Thesis im Studiengang WI Bachelor

HOCHSCHULE FURTWANGEN UNIVERSITY | HFU

Fehleranalyse im SmartCrawler-Prozess und Toolvergleich zum Erstellen einer Logo-oder Personenwiedererkennung mithilfe von Machine Learning

Thesis

zur Erlangung des Grades

Bachelor of Science

im Studiengang Wirtschaftsinformatik

an der Fakultät Wirtschaftsinformatik

der Hochschule Furtwangen University

vorgelegt von

Alexander Gerling

Referenten:	Prof. Dr. Heß
	Dr. Patrick Ndjiki-Nya
Eingereicht am:	28. Februar 2017

Abstract

Der SmartCrawler ist ein auf Java basierendes Programm, das von dem Unternehmen PAMA Technologies GmbH verwendet wird, um Bilder aus dem Internet zu laden. Diese Bilder werden im Anschluss daran benutzt, um bestimmte Personen oder Gegenstände wiederzuerkennen. Beim Herunterladen der Bilder, kommt es zu verschiedenen Fehlern, deren Ursache untersucht wurde. Daraufhin sind die Fehlerursachen benannt und der Zusammenhang erklärt worden. Die beschriebenen Fehler beim Herunterladen der Bilder, müssen behoben werden, um ein Modell zur Wiedererkennung bestmöglich zu trainieren.

In meiner Arbeit sollen der SmartCrawler-Prozess verbessert und Tools für weitere Aufgabenfelder untersucht sowie getestet werden. Ziel ist, Fehlerquellen des SmartCrawler zu identifizieren und mit geeigneten Vorschlägen zu beheben. Im Anschluss werden verschiedene Technologien in Bezug auf Gesichtserkennung, Bildbeschreibung und Machine-Learning-Tools getestet. Sie werden anhand von Beispielen zur Einsatzmöglichkeit von Gesichtserkennung und Bildbeschreibung dargestellt. Im Verlauf der Arbeit wird auch das Machine-Learning-Tool vorgestellt und exemplarisch hinterlegt. Da es sich hier um eine praktische Arbeit handelt, werden verschiedene Tools auf ihre Leistungen getestet und bewertet. Als Ergebnis wird dem Leser ein resultierender Workflow präsentiert. Der Workflow enthält den SmartCrawler-Prozess und das anschließende Training eines Modells mit CNTK.

Ein weiteres Ziel ist, dem Leser Möglichkeiten zu bieten, auf dieser Arbeit aufzubauen. Es soll für den Leser möglich sein, die richtige Anwendung für seine Bedürfnisse zu finden und eventuell eigene Projekte auf Basis dieser Arbeit zu erstellen. Weiterhin kann auch der Einsatz von CNTK vom Leser in eigenen Bereichen geprüft werden.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

Abkürzung	Ausgeschriebene Abkürzung
API	Application Programming Interface
BCV	Bildverarbeitung und Computer Vision
CNN	Convolutional Neural Networks
CNTK	Computational Network Toolkit
CPU	Central Processing Unit
GPU	Graphics Processing Unit
IT	Information Technology
kB	Kilobytes
MSO	Mathematik, Statistik und Optimierung
pgm	Portable Graymap
RNN	Recurrent neural networks
ROI	Region of Interest

1 Einleitung

Etwas zu sehen und daraus zu lernen, ist für Menschen in der Regel ein einfacher Prozess. Sie sehen ein Motorrad und ob es ein blaues oder rotes Motorrad ist, spielt keine Rolle. Menschen können richtig zuordnen beziehungsweise wiedergeben, dass es sich um ein Motorrad handelt. Nun stellt sich aber die Frage: Wie wird einer Maschine beigebracht, etwas zu tun, das für Menschen das Natürlichste überhaupt ist? Die Bereiche Machine Learning und Computer Vision sind verbreitete Themen in der IT, diese beiden Themengebiete haben jedoch mit der Leistungssteigerung der GPU-Entwicklung stark an Relevanz gewonnen. Idealerweise wird einer Maschine beigebracht, ein Objekt richtig wiederzuerkennen und mitzuteilen, was oder wer genau das ist. Dieses erlernte Wissen soll genutzt werden, um beispielsweise Personen auf Bildern (wieder-)erkennen oder Produkte einer Firma benennen zu können. Durch eine Technologie, die das menschliche Gehirn imitiert, kann eine leistungsstarke und lernfähige Lösung angeboten werden. Ein treffendes Szenario wäre, das Erkennen eines Produktes anhand eines Bildes und die darauf folgende Meldung an den Benutzer, um was es sich für ein Produkt oder um welche Marke handelt. Dem Benutzer könnten so relevante Produktdetails weitergegeben werden, um die richtige Wahl beim Einkaufen zu treffen. Ein weiteres Beispiel wäre, das Erkennen von bekannten Personen wie Barack Obama oder Angela Merkel. Mit dieser Technik können Merkmale eines Produktes oder Person dem Benutzer zur Verfügung gestellt werden.

1.1 Ziel der Arbeit

In dieser Arbeit soll das Ziel verfolgt werden, den SmartCrawler-Prozess zu verbessern und Störungen zu entfernen beziehungsweise den Prozess zu optimieren. Der SmartCrawler-Prozess bildet die Grundlage, die für das spätere Training eines Modells zur Wiedererkennung von Personen oder Objekten benötigt wird. Dabei müssen irrelevante Bilder herausgefiltert und die Suchkriterien verfeinert werden, um eine Grundlage mit relevanten Bildern zu schaffen. Das Modell soll eine möglichst hohe Genauigkeit bei der Wiedererkennung von Bildern vorweisen können. Ein gelungener Abschluss wäre, diesen Prozess zu automatisieren.

1.2 Vorgehensweise

Relevant ist, zunächst herauszufinden, welche Störungen vorliegen und welche Probleme beim Laden der Bilder entstehen. Deshalb wird zunächst mit dem SmartCrawler nach Begriffen

gesucht, um ein Fundament für die folgenden Schritte zu schaffen. Die geladenen Bilder müssen daraufhin analysiert werden, um mögliche Fehlerquellen zu finden. Nach dem Analysieren der Bilder werden Technologien und Verbesserungspotenziale herausgesucht, um die Störungen zu beheben. Tools unterschiedlicher Hersteller werden miteinander verglichen. Über den sinnvollen Einsatz von Open-Source-Tools soll nachgedacht werden. Anschließend wird ein Vergleich realisiert, um sich für ein Tool oder eine Kombination aus verschiedenen Tools zu entscheiden.

1.3 Machine Learning als Werkzeug

Die Frage, wie einer Maschine beizubringen ist, wie sie lernen soll, trägt bereits weitere Fragestellungen in sich: Was ist maschinelles Lernen? Was braucht eine Maschine, um zu lernen? Wie funktioniert maschinelles Lernen?

Die Definition von maschinellem Lernen lautet:

> *„Anwendung und Erforschung von Verfahren, durch die Computersysteme befähigt werden, selbstständig Wissen aufzunehmen und zu erweitern, um ein gegebenes Problem besser lösen zu können als vorher.“*[1]

Daraus lässt sich schließen, dass Maschinen unter bestimmten Bedingungen fähig sind, selbstständig zu lernen und sich bei wiederholtem Anwenden von Verfahren weiter zu verbessern. Notwendig zum Lernen sind Informationen, die die Maschine anhand von Daten bekommt. Daten sind zwingend für den Lernprozess einer Maschine erforderlich. Anhand dieser Daten lassen sich Trainings- und Testdurchläufe gestalten, um Modelle zu erstellen. Hierbei sollte darauf geachtet werden, dass das Modell nicht in den Zustand „Underfitting“ oder „Overfitting“ gerät. Beim Overfitting wird von einem Modell gesprochen, das beispielsweise zu viele unbedeutende Variablen besitzt. Im Gegenzug wird es als Underfitting bezeichnet, wenn relevante Variablen nicht benutzt werden. Diese Zustände werden hervorgehoben, wenn beim Training der Daten nicht auf die Generalisierung geachtet wird.[2]

[1] Springer Gabler Verlag (Hg.): Definition » maschinelles Lernen « | Gabler Wirtschaftslexikon. Online verfügbar unter http://wirtschaftslexikon.gabler.de/Definition/maschinelles-lernen.html, zuletzt geprüft am 27.01.2017 .

[2] Vgl. Görz, Günther; Schneeberger, Josef; Schmid, Ute (2014): Handbuch der künstlichen Intelligenz. 5., überarb. und aktualisierte Aufl. München: Oldenbourg, S 387.

Zu den möglichen Verfahren von Machine Learning gehören beispielsweise:[3]

- Decision Trees
- Artificial Neural Networks
- Bayesian Learning
- Clustering
- Support Vector Machine

Ein bedeutender wiedererweckter Teil von Machine Learning sind neuronale Netzwerke. Diese haben durch die immense Leistungssteigerung der GPU (Graphics Processing Unit) in den vergangenen Jahren wieder Aufwind bekommen und an Bedeutung zugenommen.

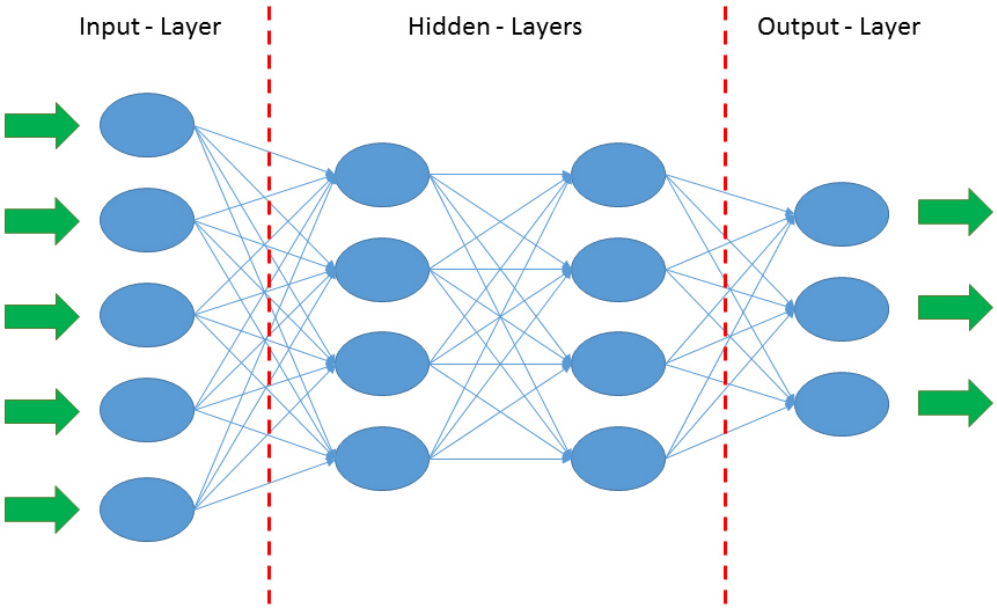

Abbildung 1: Neuronales Netz[4]

Ein neuronales Netz stellt eine abstrakte Informationsverarbeitung dar, wie sie vergleichbar im menschlichen Nervensystem abläuft. Das neuronale Netz besteht aus mehreren miteinander verknüpften Neuronen, die in unterschiedlichen Schichten liegen. Der Informationsaustausch findet zwischen den unterschiedlichen Schichten statt. Die Anzahl von Schichten und somit der

[3] Vgl. Görz, Günther; Schneeberger, Josef; Schmid, Ute (2014): Handbuch der künstlichen Intelligenz. 5., überarb. und aktualisierte Aufl. München: Oldenbourg, S 413-460.
[4] Vgl. Mitchell, Tom M. (20]10): Machine learning. International ed., [Reprint.]. New York, NY: McGraw-Hill (McGraw-Hill series in computer science), S 107.

Neuronen wird vom Designer beziehungsweise vom Programmierer je nach Wunsch oder Notwendigkeit variabel gestaltet.[5][6]

Die Schichten werden folgendermaßen geteilt:

Input-Layer
Besitzt die Eingaben und leitet an das Netz weiter.

Hidden-Layer(s)
Dient zur Informationsverarbeitung innerhalb des Netzes.

Output-Layer
Ausgabe des Netzes.

Bei neuronalen Netzen lässt sich das Lernen in Supervised Learning und in Unsupervised Learning unterteilen.[7] Beim Supervised Learning werden dem neuronalen Netz die Daten und ein oder mehrere Classifier bereitgestellt. Übliche neuronale Netze wären dafür die Recurrent Neural Networks(RNN) oder Convolutional Neural Networks(CNN).

Beim Unsupervised Learning werden dem Netz nur Daten bereitgestellt. Somit ist es dem neuronalen Netz selbst überlassen, was es damit macht.

Ein neuronales Netz kann auf verschiedene Arten lernen:[8]

- Entwicklung neuer Verbindungen beziehungsweise Löschen von bestehenden Verbindungen
- Änderung in der Gewichtung verschiedener Neuronen
- Schwellenwerte der Neuronen werden angepasst
- Hinzufügen oder Löschung von Neuronen

[5] Vgl. Kruse, Rudolf; Borgelt, Christian; Braune, Christian; Klawonn, Frank; Moewes, Christian; Steinbrecher, Matthias (2015): Computational Intelligence. Eine methodische Einführung in künstliche neuronale Netze, evolutionäre Algorithmen, Fuzzy-Systeme und Bayes-Netze. 2., überarbeitete und erweiterte Auflage. Wiesbaden: Springer Vieweg (Computational Intelligence), S 44.
[6] Vgl. Mitchell, Tom M. (20]10): Machine learning. International ed., [Reprint.]. New York, NY: McGraw-Hill (McGraw-Hill series in computer science), S 107.
[7] Vgl. Görz, Günther; Schneeberger, Josef; Schmid, Ute (2014): Handbuch der künstlichen Intelligenz. 5., überarb. und aktualisierte Aufl. München: Oldenbourg, S 377-386.
[8] Vgl. Görz, Günther; Schneeberger, Josef; Schmid, Ute (2014): Handbuch der künstlichen Intelligenz. 5., überarb. und aktualisierte Aufl. München: Oldenbourg, S 365.

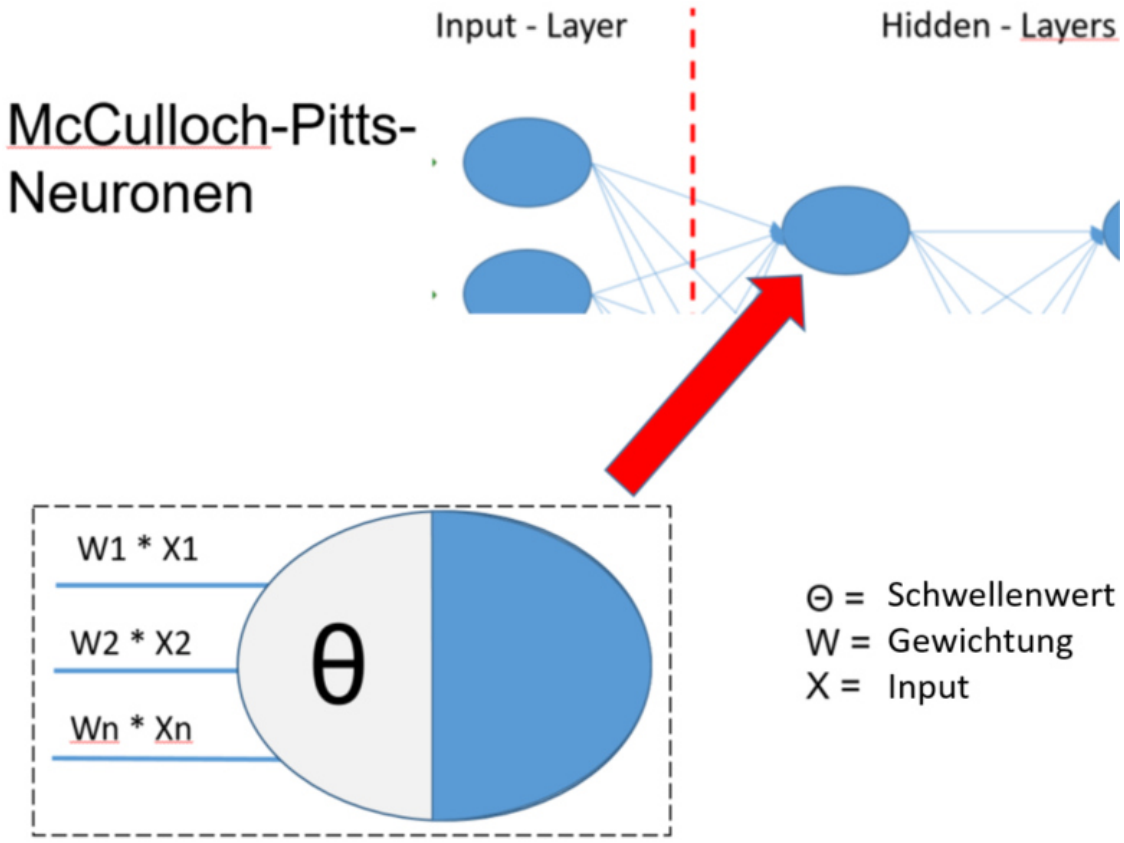

Abbildung 2: McCulloch-Pitts-Neuron[9]

Anhand der Neuronen wird das „Lernen“ erst möglich gemacht. Diese können durch Anpassung der Gewichtung und der Schwellenwerte das Lernverhalten beeinflussen.[10]

[9]Vgl. Mitchell, Tom M. (20]10): Machine learning. International ed., [Reprint.]. New York, NY: McGraw-Hill (McGraw-Hill series in computer science) S 87.
[10] Vgl. Görz, Günther; Schneeberger, Josef; Schmid, Ute (2014): Handbuch der künstlichen Intelligenz. 5., überarb. und aktualisierte Aufl. München: Oldenbourg, S 365.

2 Problemstellung

Das Unternehmen PAMA Technologies hat den SmartCrawler auf Basis von Java entwickelt und dieser erlaubt es dem Benutzer, Bilder mit bestimmten Inhalten aus dem Internet zu laden. Das Besondere an diesem Programm ist, dass es dem Suchbegriff auch Keywords hinzufügt, um die Reichweite der geladenen Bilder zu erhöhen. Als Beispiel wird mit dem SmartCrawler nach dem Logo des Unternehmens Puma gesucht. Resultierende Suchanfragen wären hier „Puma Logo“, „Puma Logo T-Shirt“, „Puma Logo Jacke“, „Puma Logo Schuhe“. Diese Erweiterung des Suchfeldes ist sinnvoll, da in weiteren Schritten ein Modell mit diesen Bildern trainiert und getestet wird. Die Problematik besteht darin, dass nur wenige der geladenen Bilder für das anschließende Training verwendbar sind. Somit muss der Workflow auf Fehlerquellen analysiert werden. Der SmartCrawler-Prozess wird in diesem Abschnitt analysiert, um Probleme und Störungen aufzuzeigen. Diese können danach behoben werden.

2.1 SmartCrawler-Workflow

Der SmartCrawler startet mit einer Benutzereingabe. Beispielsweise lautet ein Befehl „java –jar smartCrawler.jar puma logo“. Nach der Eingabe des Benutzers wird der Internetbrowser geöffnet und die Seite www.keywordtool.io aufgerufen. Diese Website bietet dem Benutzer die Möglichkeit, Keywords an den Suchbegriff anzuheften. Nach dem Laden der Keywords werden diese für spätere Anfragen in eine Datenbank geschrieben. Mithilfe der Keywords startet der Download auf unterschiedlichen Suchmaschinen. Die geladenen Bilder befinden sich nach dem Abschluss des Prozesses auf dem Server. Auch die Datenbank hat einen Eintrag mit dem Suchbegriff und der Anzahl der geladenen Bilder erhalten.

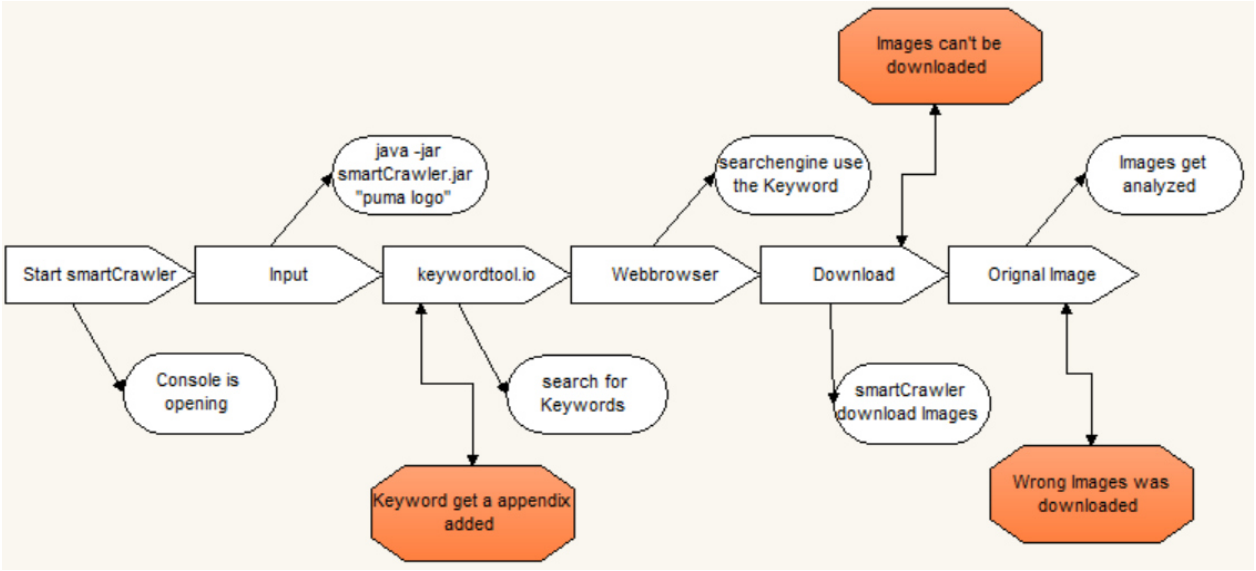

Abbildung 3: SmartCrawler-Workflow

Quelle: eigene Darstellung

2.2 Auftretende Probleme und Fehlerursachen

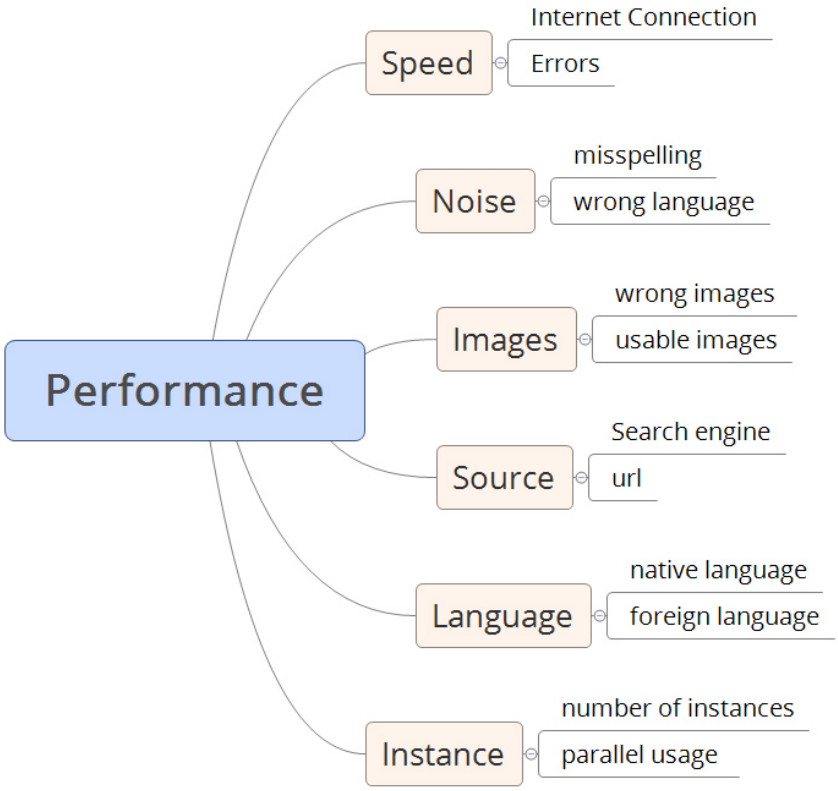

Abbildung 4: SmartCrawler-Mindmap

Quelle: eigene Darstellung

Die Abbildung 4 zeigt eine Aufspaltung der Performance in ihre relevanten Kriterien. So wird im Vorfeld herausgefunden, welche Probleme beim Laden der Bilder entstehen können. Störungen entstehen unter anderem durch Rechtschreibfehler wie „CKU“ anstatt „CDU“. Die Wahl der richtigen Sprache ist hier mindestens genauso relevant. Dementsprechend wäre es richtig, nach CDU zu suchen, wenn die Christlich Demokratische Union aus Deutschland gemeint ist, aber genauso richtig, wenn „CKU“ für „Zentrum für Weiterbildung“ in Polen stehen soll.

Beim Punkt Images kann eine Störungsquelle darin liegen, dass generell falsche Bilder geladen werden. Beispielsweise werden Logos von Puma gesucht und das Logo des 1. FC Köln mit einem Ziegenbock kommt zurück. Zu diesem Punkt gehören auch Bilder, die zwar inhaltlich etwas mit dem Begriff zu tun haben, aber nicht das gewünschte Resultat liefern. Wenn Bilder vom Logo des Oscars benötigt werden, jedoch Bilder des roten Teppichs mit Prominenten geladen werden, sind diese ebenso als falsch zu bewerten.

Ein weiterer Kritikpunkt liegt bei der parallelen Nutzung von mehreren SmartCrawler-Instanzen. In der Praxis durften nicht mehr als vier Instanzen aufgerufen werden, da es sonst zu Problemen mit dem Internetbrowser geführt hat. Ebenso war die Anzahl der Reiter (Tabs) Internetbrowsers begrenzt verwaltbar.

Ein bedeutender Punkt beim Laden der Bilder ist die Wahl der richtigen Suchmaschine. Vor der Suchanfrage ist es entscheidend, zu wissen, ob es sich um einen inländischen oder ausländischen Begriff handelt. Hier unterscheiden sich die Resultate bei Verwendung einer fremdsprachigen Suchmaschine in Verbindung mit einem deutschen Suchbegriff deutlich. In der nachfolgenden Abbildung 5 ist zu sehen, dass nach dem Begriff „CSU“ gesucht wurde. Hier kamen die Suchmaschinen google.de (links im Bild) und google.co.in (rechts im Bild) zum Einsatz. Die Ergebnisse könnten kaum unterschiedlicher sein. Somit sollte diesem Kritikpunkt wesentlich mehr Bedeutung zukommen.

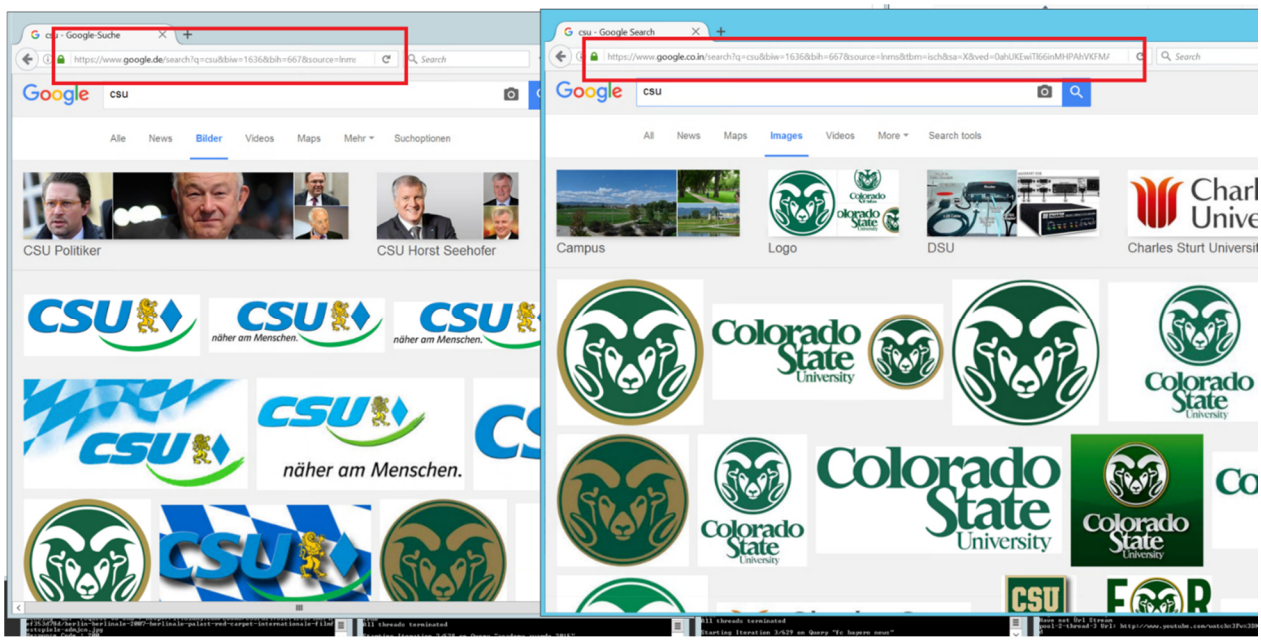

Abbildung 5: Suchmaschinenvergleich

Quelle: eigene Darstellung

2.3 Ergebnisse und mögliches Verbesserungspotenzial

Für die Analyse der Ergbenisse wurden zwei Durchläufe mit dem SmartCrawler durchgeführt. Im Vorfeld ist anzumerken, dass der erste Durchlauf mit eventuellen Fehleingaben stattgefunden hat. Mit Fehleingaben ist gemeint, dass im ersten Durchlauf keine saubere Eingabestruktur vorhanden war. Es sollte getestet werden, wie der SmartCrawler auf Eingaben wie „polizei“, „Polizei“ oder „POLIZEI“ reagiert.

Die Suche mit dem SmartCrawler wurde in vier Kategorien zu je fünf Themen unterteilt:

- Politik
- Sport
- Arts and Entertaiment
- Brands (Logos)

In den folgenden Abbildungen 6 und 7 sind die Anzahl der geladenen und der verwendbaren Bilder sowie die daraus resultierende Genauigkeit zu sehen. Die meisten Ergebnisse fallen nicht besonders gut aus.

Politik	Images	usable images	accurat %	Sport	Images	usable images	accurat %
Polizei	18	0	0,0	FIFA	2	2	100,0
CSU	6	0	0,0	IOC/ IOK	1	1	100,0
IG Metall	186	3	0,0	Olympische Sommerspiele 2016	2	2	100,0
Siegel des Präsidenten der Vereinigten Staaten	2	0	0,0	Real Madrid	78	0	0,0
Flagge: Syrien	20	0	0,0	FC Bayern München	20	4	20,0
Arts and Entertainment	**Images**	**usable images**	**accurat %**	Brands	Images	usable images	accurat %
Walt Disney Pictures	55	7	12,7	MasterCard	7	6	85,7
Academy Awards (Oscars)	80	5	6,3	Audi	31	20	64,5
Internationale Filmfestspiele Berlin (Berlinale)	9	1	11,1	Nike	144	58	40,3
Eurovision Song Contest	68	13	19,1	Burger King	54	2	3,7
n-tv	29	0	0,0	Apple	245	181	73,9

Abbildung 6: SmartCrawler erster Durchlauf

Quelle: eigene Darstellung

Politik	Images	usable images	accurat %	Sport	Images	usable images	accurat %
Polizei	23	0	0,0	FIFA	302	27	8,9
CSU	193	0	0,0	IOC/ IOK	16	1	6,3
IG Metall	720	31	4,3	Olympische Sommerspiele 2016	123	1	0,8
Siegel des Präsidenten der Vereinigten Staaten	29	0	0,0	Real Madrid	1251	80	6,4
Flagge: Syrien	14	7	50,0	FC Bayern München	384	32	8,3
Arts and Entertainment	**Images**	**usable images**	**accurat %**	Brands	Images	usable images	accurat %
Walt Disney Pictures	244	17	7,0	MasterCard	160	67	41,9
Academy Awards (Oscars)	578	58	10,0	Audi	105	80	76,2
Internationale Filmfestspiele Berlin (Berlinale)	41	0	0,0	Nike	595	275	46,2
Eurovision Song Contest	122	23	18,9	Burger King	38	7	18,4
n-tv	29	3	10,3	Apple	1229	486	39,5

Abbildung 7: SmartCrawler zweiter Durchlauf

Quelle: eigene Darstellung

3 Technologie

In diesem Kapitel werden verschiedene Technologien vorgestellt, die in der Thesis verwendet und auf ihre möglichen Leistungen getestet wurden. Die Technologien haben ihren Fokus dabei auf Face Detection gerichtet. Im Zeitraum der praktischen Tests war noch nicht klar, in welche Richtung die Thesis gehen sollte. Nach Zwischenauswertungen der Besprechungsmeetings wurde die Auswahl der eingesetzten Tools überarbeitet und neu entschieden. Dabei war herauszufinden, welche Möglichkeiten schon bestehen und für diese Aufgabe nutzbar sind. Somit ist die Auswahl der Tools im Lauf der praktischen Arbeit entstanden.

3.1 Programmiersprachen

In diesem Abschnitt werden die verwendeten Programmiersprachen erläutert. Die bereitgestellten Programmcodebeispiele werden in den folgenden Abschnitten hauptsächlich in den populären Sprachen Python und Java verwendet. Die Programmcodebeispiele sind von den jeweiligen Plattformen entnommen oder durch Mitglieder der Community bereitgestellt worden.

3.1.1 Python

Python ist eine universelle, objektorientierte Open-Source-Programmiersprache. Weiterhin kann Python als Skript oder auch als eigenständiges Programm aufgerufen werden. Python ist als Programmiersprache zur Optimierung der Komponentenintegration, der Programm Portabilität und der Produktivität von Entwicklern entworfen worden und kann auf den meisten Plattformen wie Microsoft, Linux, Unix etc. ohne Probleme eingesetzt werden.[11] Durch die einfache Struktur, die Python besitzt, hat sie sich als Standardprogrammiersprache im Computer-Vision-Bereich etabliert.

3.1.2 Java

Java ist eine plattformunabhängige objektorientierte Programmiersprache, die im Jahr 1995 von Sun Microsystems veröffentlicht wurde. Als universelle Programmiersprache findet Java deshalb Verwendung auf Clients, Servern oder auch Mobile Devices. Somit ist Java ein

[11] Vgl. Lutz, Mark; Schulten, Lars (2010): Python. Kurz & gut. 4. Aufl. Köln: O'Reilly (O'Reillys Taschenbibliothek)

etablierter Standard in der Entwicklung von Anwendungen. [12] Dabei ist Java als Programmiersprache recht einfach und robust gehalten und hat aus den Erfahrungen von Smalltalk, C und C++ profitiert.[13]

3.2 OpenCV

OpenCV (Open Source Computer Vision) wurde von Intel entwickelt und wird als Open-Source-Bibliothek plattformübergreifend für Bildverarbeitung verwendet. OpenCV hat sich als Standardwerkzeug im Bereich Computer Vision etabliert. Praktisch dabei ist, dass OpenCV ohne Probleme in verschiedene Programmiersprachen integriert werden kann. Die erste Version von OpenCV wurde im Jahre 2000 veröffentlicht. Durch die dahinterstehende Community hat OpenCV viel an Funktionalität dazugewonnen.[14] Die aktuelle OpenCV-Version 3.2 weist inhaltliche und rechtliche Unterschiede zur Version 2.4.13 auf. Deshalb sollte vor Gebrauch von OpenCV die nötige Funktionalität geprüft und anhand dessen die eingesetzte Version bestimmt werden. Vorweg muss gesagt werden, das CNTK die OpenCV Version 3.1 voraussetzt. Bei fehlender Funktionalität, die bei der vorherigen Version vorlag, kann eine eigene Version von OpenCV erstellt werden. Hierzu können benötigte Funktionalitäten aus beiden Versionen zu einer kombiniert werden.

3.3 Gesichtserkennung und Problematik

Die Erkennung von Gesichtern in Bildern ist kein neues Themengebiet im Bereich Computer Vision, jedoch eines, das mit einigen Problemen behaftet ist. Die Erkennung einer oder mehrerer Personen hinter einer weißen Leinwand gehört sicherlich zu den Idealfällen. Bei Bildern, die durch Hintergründe und zusätzliche Objekte erweitert werden, fällt die Erkennung jedoch um einiges schwerer. Problematisch wird es in dunklen Umgebungen oder durch zu stark beeinflussende Lichtverhältnisse. Die Hautfarbe des Menschen spielt dabei auch eine nicht zu unterschätzende Rolle, weil diese in viele Farbtöne unterteilt ist. Somit ist eine simple

[12] Vgl. Abts, Dietmar (2016): Grundkurs JAVA. Von den Grundlagen bis zu Datenbank- und Netzanwendungen. 9., überarbeitete und erweiterte Auflage. Wiesbaden: Springer Vieweg (Lehrbuch)
[13] Vgl. Java Timeline. Online verfügbar unter http://oracle.com.edgesuite.net/timeline/java/, zuletzt geprüft am 10.02.2017
[14] Vgl. Garcia, Gloria Bueno; Gracia, Ismael Serrano; Aranda, Jose Lius Espinosa; Salido Tercero, Jesus; Enano, Noelia Vallez; Suarez, Oscar Deniz (2015): Learning image processing with OpenCV. Exploit the amazing features of OpenCV to create powerful image processing applications through easy-to-follow examples. Birmingham, UK: Pakct Publishing (Community experience distilled).

Erkennung anhand eines speziellen Farbtons nicht möglich.[15] Spezielle Eigenschaften des Gesichtes können bei der Erkennung helfen. Markante Gesichtsstellen wie Augen, Nase, Mund oder Augenbrauen können hier überaus hilfreich sein.

Die Viola-Jones-Methode, die als einer der ersten performanten Ansätze zur Gesichtserkennung funktioniert, kann Gesichter, die direkt in die Kamera blicken, gut erkennen.[16] Trotzdem weist diese Methode Probleme bei Drehungen des Kopfes auf. Es bestehen Lösungsansätze, die diese Problematik mit Kombinationen aus Gesichtserkennung, Position des Kopfes und Landmarken des Gesichtes in einem Framework lösen.[17] Die Gesichtserkennung ist auch mit einem CNN möglich. Diese Technik wurde durch Verbesserungen der Technik wieder in Betracht gezogen. Hier bestimmt jedoch die Maschine selbst, welche Gesichtsmerkmale zum Lernen des Gesichtsmusters benutzt werden.

3.4 Google Cloud Vision API

Google bietet eine ganze Palette an Schnittstellen für Entwickler an.[18] Positiv fällt auf, dass Google den Entwicklern von Anfang an eine Auswahl an Schnittstellen für verschiedenen Programmiersprachen anbietet. Die Entscheidung fiel auf Python, da die Programmiersprache praktikabel und unkompliziert einsetzbar ist. Auch wurden Skripte mit Java getestet. Weiterhin bietet Google mehrere Samples und Tutorials für die Benutzer an. Somit ist der Einstieg für den Benutzer leichter und der vorgegebene Code kann auch auf die jeweiligen Bedürfnisse angepasst werden. Um den Service von Google zu nutzen, muss zuerst eine Registrierung auf der Google-Plattform vorgenommen werden. Auch die jeweiligen Kontodaten müssen übermittelt werden, was automatisch heißt, dass dieser Service mit Kosten verbunden ist. Sobald die Formalitäten abgeschlossen sind, muss eine Authentifikation mit der Google-Plattform vorgenommen werden. Hier wird dem Benutzer per Anleitung vorgeführt, wie der API Key für die Nutzung der Google-Dienste zu beschaffen ist. Für jeden Test mit der API wird der erstellte Key zwangsläufig verwendet.

[15] Vgl. Advances in Face Detection and Facial Image Analysis (2016). Cham: Springer International Publishing, S. 1.
[16] Vgl. Viola, Paul; Jones, Michael J. (2004): Robust Real-Time Face Detection. In: *International Journal of Computer Vision* 57 (2), S. 137–154. DOI: 10.1023/B:VISI.0000013087.49260.fb
[17] Vgl. Advances in Face Detection and Facial Image Analysis (2016). Cham: Springer International Publishing, S 4.
[18] Vgl. Cloud Vision API. Online verfügbar unter https://cloud.google.com/vision/, zuletzt geprüft am 27.01.2017.

Die folgenden Tests dienten dazu, die Label Detection und die Face Detection zu überprüfen, um sich ein Bild über die Qualität der Ausgaben zu machen.

3.4.1 Google Label Detection

Die vorbereiteten Java Skripte mussten nun mit den gewünschten Bildern angehängt werden. Beispielsweise wurde ein Skript in der Konsole folgendermaßen ausgeführt: „java – jar LapApp.jar nike.jpg“.

Abbildung 8: Nike-Logo[19]

Das benutzte Logo für diesen Test war ein Markenlogo von Nike.

```
root@alex-VirtualBox:/home/alex/Schreibtisch/TestApp# java -jar LapApp.jar nike.jpg
Labels for image nike.jpg:
        line (score: 0,609)
        shape (score: 0,582)
        moustache (score: 0,565)
```

Abbildung 9: Analyse Nike-Logo

Quelle: eigene Darstellung

Hier wird die Unterteilung des Nike-Logos in die Kategorien line, shape und moustache mit der jeweiligen Genauigkeit erhalten. Im Skript selbst kann die Anzahl der Kategorien ohne weitere Probleme durch die Änderung der folgenden Zeile im Skript eingestellt werden: „private static final int MAX_LABELS = 3;“ Mit der Änderung der rot markierten Zahl wird die gewünschte Anzahl der Ausgaben angepasst. Zu beachten ist aber, dass durch die Erhöhung der Ausgaben die Kosten steigen. Es gibt mehrere Preisvarianten, die Google anbietet.[20] Hier ist entscheidend, wie intensiv die Funktion später genutzt wird und ob sie in Kombination mit

[19] Nike Logo. Online verfügbar unter http://news.nike.com/, zuletzt geprüft am 27.01.2017.
[20] Google Cloud Vision API Preisliste. Online verfügbar unter https://cloud.google.com/vision/pricing, zuletzt geprüft am 15.02.17.

einer weiteren Funktion steht. In Kombination mit einer weiteren Funktion steigt automatisch der Preis pro Abfrage.

Das erzielte Ergebnis anhand des Logos ist interessant, da das Nike-Logo in seine Form unterteilt und nicht als „Nike-Logo“ identifiziert wurde. Anhand von zwei weiteren Untersuchungen soll diese Bedeutung erklärt werden.

Abbildung 10: Katze[21]

In diesem Beispiel soll getestet werden, wie der Label Detector auf Tiere reagiert.

```
root@alex-VirtualBox:/home/alex/Schreibtisch/TestApp# java -jar LapApp.jar Katze.jpg
Labels for image Katze.jpg:
        cat (score: 0,986)
        mammal (score: 0,935)
        vertebrate (score: 0,921)
```

Abbildung 11: Analyse Katzenbild

Quelle: eigene Darstellung

In dem aufgeführten Bild wurde eindeutig eine Katze identifiziert und mit einer Genauigkeit von 98,6 % angegeben. Auch die zweite Kategorie „mammal“, zu Deutsch „Säugetier“, wurde präzise erkannt.

[21] Katze. Online verfügbar unter http://klaus-von-gierke.de/wp-content/uploads/Katze-Senior.jpg, zuletzt geprüft am 27.01.2017.

Der dritte Test soll aufzeigen, aus welchem Grund die Label Detection zwar gute Resultate erbringt, aber nicht unbedingt für die vorliegende Anwendung geeignet ist.

Abbildung 12: Adidas-Logo[22]

Als Ergebnis auf das Logo wurde erhalten:

- Text mit einer Genauigkeit von 91,6 %
- Logo mit einer Genauigkeit von 90,5 %

In weiteren Tests wurde deutlich, dass nicht alle Logos erkannt und diese oft in ihre Grundformen wie „text“, „font“, „line“ unterteilt wurden. Zusätzlich sind auch Test mit Personen durchgeführt worden. Interessant an diesen Tests war, dass als Resultat eine recht gute Situationsbeschreibung entstand.

[22] Adidas Logo. Online verfügbar unter https://upload.wikimedia.org/wikipedia/commons/thumb/d/d4/Adidas-logo.svg/2000px-Adidas-logo.svg.png, zuletzt geprüft am 27.01.2017.

Abbildung 13: Rede Bill Gates[23]

In diesem Fall wurde erkannt, dass eine Rede stattfand. Weitere Tests mit mehreren Personen sind als „team“ (85 % Genauigkeit), „community“ (67 % Genauigkeit) oder „audience“ (65 % Genauigkeit) beschrieben beziehungsweise identifiziert worden.

3.4.2 Google Face Detection

Im nächsten Schritt wurde die Face Detection überprüft. Diese Tests sind mit einer und mehreren Personen abgelaufen. Auch die Erkennung einer Handpuppe als Mensch ist geglückt, was jedoch als negativer Punkt aufgefasst werden sollte.

[23] Bill Gates Speech. Online verfügbar unter http://3.bp.blogspot.com/-HllDa0woDpY/UfAsz0KTfvI/AAAAAAAAAlw/UZ503x1eYIY/s1600/Bill+Gates+Microsoft+Will+Not+Return.jpg, zuletzt geprüft am 27.01.2017.

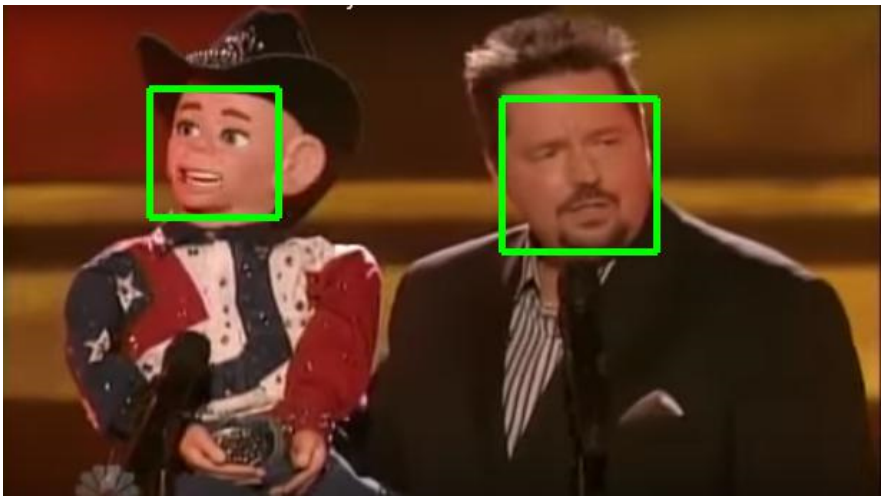

Abbildung 14: Google Face Detection

Quelle: eigene Darstellung

Das ausgeführte Script gab in der Kommandozeile aus, wie viele Personen erkannt wurden, und gab die Position der Gesichter im Bild wieder. Somit wurden in der Abbildung 14 zwei Personen erkannt.

3.4.3 Fazit zu Google Cloud Vision API

Die Qualität der Ergebnisse war erstaunlich gut, besonders für eine Person, die zunächst keinen vorherigen Kontakt mit dieser Thematik hatte. Ebenfalls positiv zu sehen ist, dass ein Script mit Kombinationen aus mehreren Detektoren geschrieben werden könnte. Somit bietet Google ein solides Tool und lässt Potenzial für eigene Anpassungen offen. Jedoch sollte beachtet werden, dass bei intensiver Benutzung der API schnell hohe Kosten entstehen können. Auch waren die Ergebnisse für einen Einsatz eher durchschnittlich zu bewerten. Entscheidend sind die Abhängigkeit zu Google als Dienstleister und das Outsourcen von Kompetenzen. Sobald eine Software mit dieser API entwickelt wurde, ist mit ständig anfallenden Kosten zu rechnen.

3.5 Matlab

Matlab ist eine kommerzielle Software, die es dem Benutzer vereinfacht, mathematische Probleme zu lösen und Ergebnisse grafisch darzustellen.[24] Angeboten wird diese Software vom US-Unternehmen MathWorks. Matlab ist nach einer kleinen Einstiegsphase gut für Benutzer geeignet, die geringe Programmierkenntnisse vorweisen. Durch die Eingaben in der Kommandozeile werden Befehle ausgeführt. Auch die Programmierung wird über die Kommandozeile getätigt.

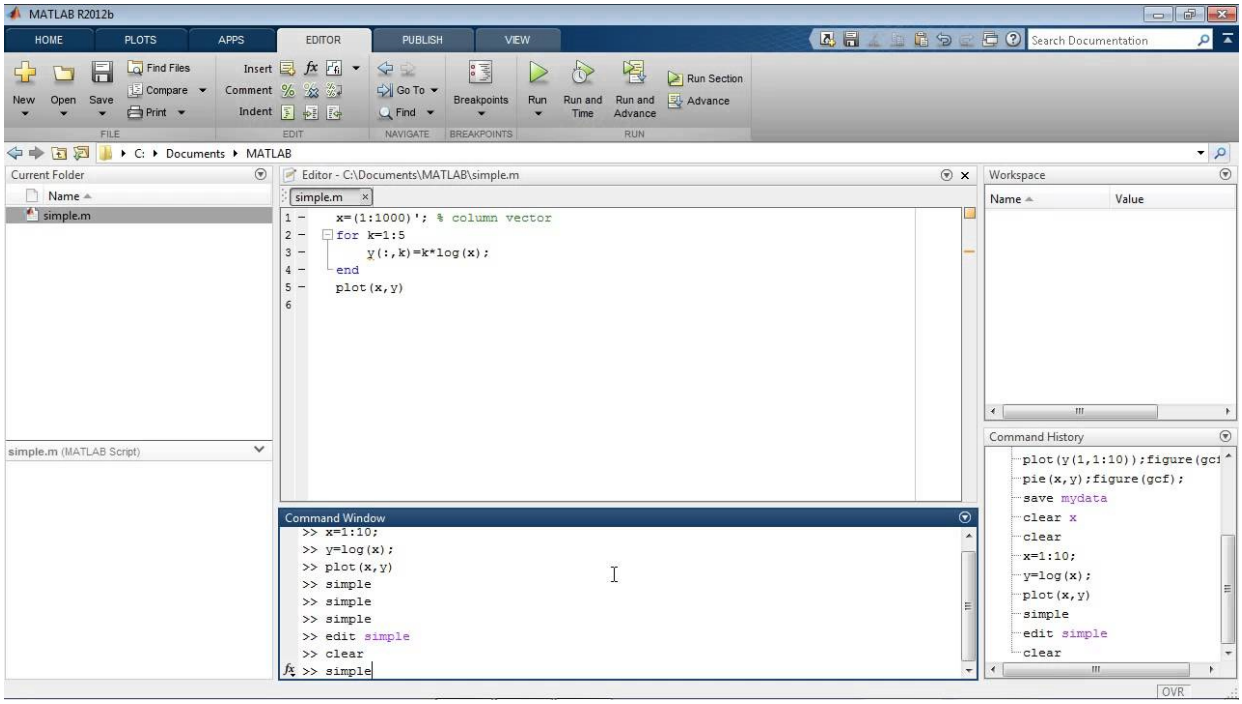

Abbildung 15: Matlab-Programm[25]

Überaus positiv an Matlab ist die Erweiterung des Standardprogrammes durch Add-on-Produkte. Interessante Erweiterungen für diese Thematik sind:

- Mathematik, Statistik und Optimierung (folgend MSO genannt)
- Bildverarbeitung und Computer Vision (folgend BCV genannt)

Inhalte im MSO sind unter anderem die Neural Network Toolbox, um neuronale Netzwerke zu erstellen, zu trainieren und zu simulieren, sowie die Statistics and Machine Learning Toolbox,

[24] Vgl. Matlab. Online verfügbar unter https://de.mathworks.com/products/matlab.html, zuletzt geprüft am 27.01.2017.
[25] Matlab. Online verfügbar unter https://i.ytimg.com/vi/pRsGM7H91VY/maxresdefault.jpg, zuletzt geprüft am 27.01.2017.

um Analyse und Modellierung von Daten mithilfe von Statistik und maschinellem Lernen zu ermöglichen.

Das Add-on-BCV besteht aus folgenden Inhalten:

- Image Processing Toolbox (zuständig für Bildverarbeitung, Bildanalyse und Algorithmen Entwicklung)
- Computer Vision System Toolbox (zuständig für die Auslegung und Simulierung von Computer Vision Systemen und Videoverarbeitung)
- Mapping Toolbox (zuständig für die Analysierung und Visualisierung geografischer Informationen

Mit diesen Erweiterungen bietet Matlab eine mächtige Software, die den notwendigen komplexen Anforderungen gerecht wird. Zusätzlich bietet Matlab die Möglichkeit, Prozesse zu parallelisieren, um schneller Ergebnisse zu realisieren.

Ein nicht zu unterschätzender Punkt bei Matlab ist die Community der Benutzer. In einigen Fällen gab es Kollaborationen in Projekten und das Teilen des geschriebenen Codes sowie das Unterstützen anderer Benutzer als wesentlicher Bestandteil. Somit lassen sich bei Problemen die Mitglieder der Community um Rat bitten und publizierte Codes anderer Benutzer für eigene Projekte verwenden, was zur erheblichen Effizienzsteigerung des Programmes führt.

Um die Matlab-Software zu testen, sind folgende Punkte untersucht worden:

- Face Detection
- Face Recognition
- Template Matching
- Templatesuche in einer Bilderabfolge

3.5.1 Face Detection

Zunächst wird die Face Detection überprüft.

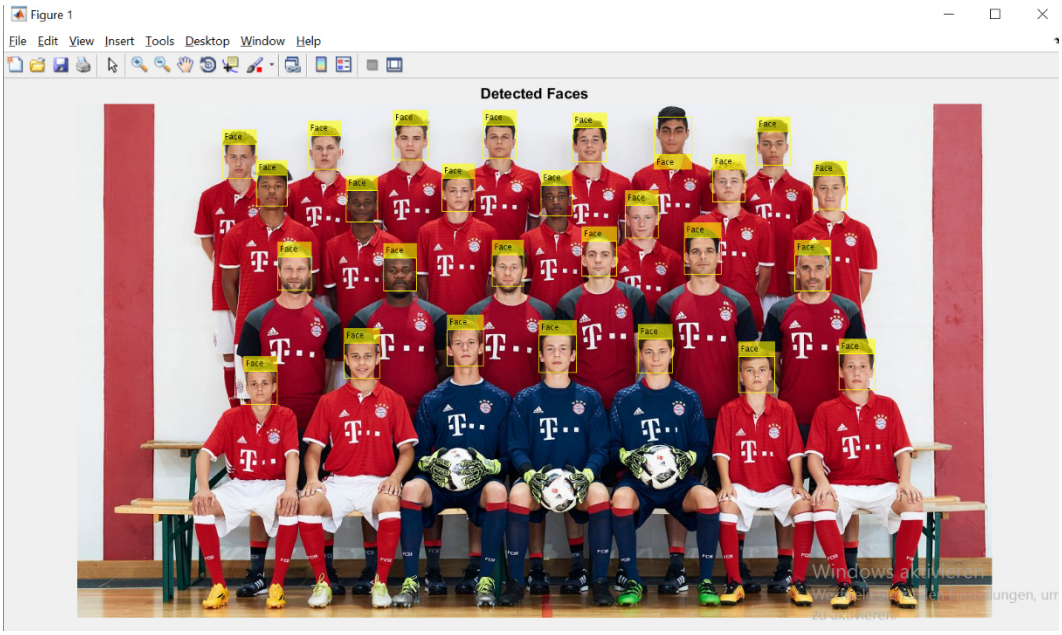

Abbildung 16: Matlab Face Detection

Quelle: eigene Darstellung

Bei Bildern mit einer hohen Auflösung hat Matlab keine Schwierigkeiten, alle Gesichter der Fußballspieler zu erkennen. Hier ist die Auflösung ein entscheidender Faktor. Je höher die Auflösung der Bilder, desto präzisere Ergebnisse können erzielt werden.

3.5.2 Face Recognition

Bei der Face Recognition soll ein Gesicht in einem Bild mit einer oder mehreren Personen wiedererkannt werden. Hierfür werden einige Bilder einer Person aus unterschiedlichen Positionen benötigt.

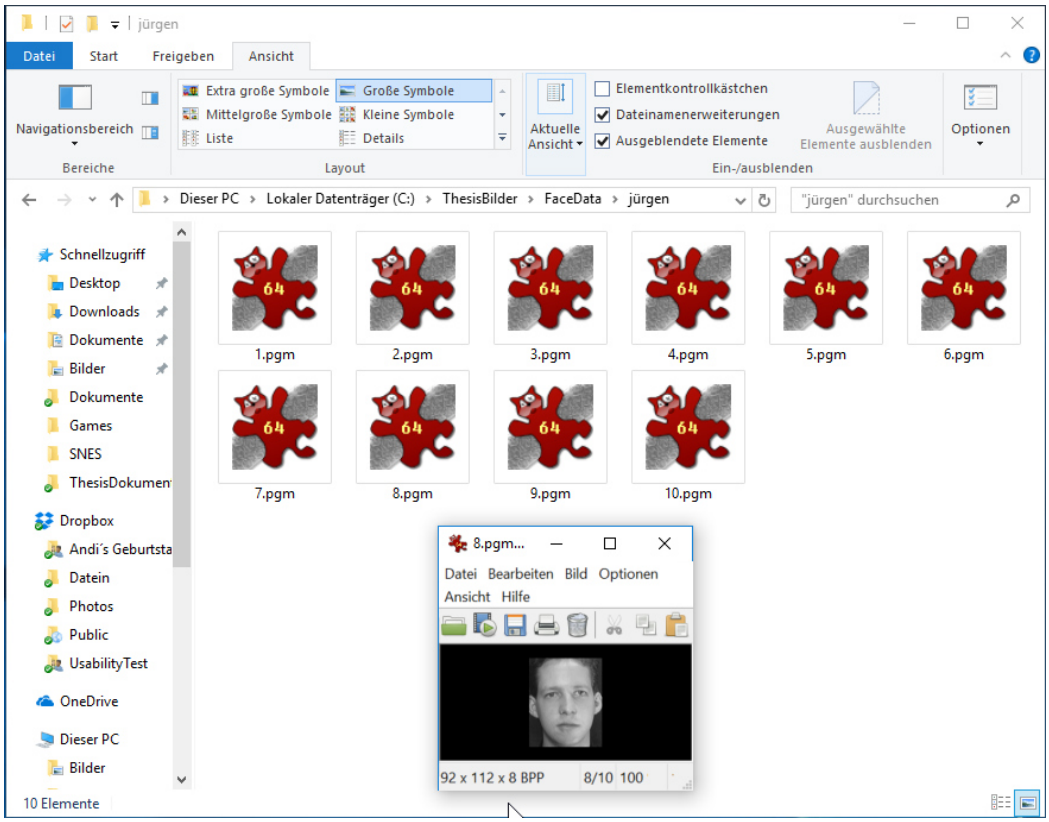

Abbildung 17: Matlab Face Recognition

Quelle: eigene Darstellung

In diesem Beispiel werden zehn Bilder einer Person mit identischer Größe von 10 kB verwendet, um das Modell zu trainieren. Parallel lassen sich weitere Personen mit trainieren. Es muss lediglich drauf geachtet werden, dass eine saubere Struktur angelegt wird. Eine mögliche Struktur wäre „-> Ordner1 – Ordner 40“ mit jeweils 10 Bildern, die das Format „pgm“ nutzen. Mit dieser Struktur kann das Modell auf 40 unterschiedliche Personen trainiert und anschließend auf die Genauigkeit getestet werden.

In der folgenden Darstellung gibt es jeweils zwei Eingabebilder (Query Face) einer Person, um zu sehen, was als resultierendes Ergebnis zurückgegeben wird. Gut zu erkennen ist, dass die Person trotz einer Drehung des Kopfes der richtigen Klasse zugeordnet werden kann. Nur bei einem der insgesamt 20 Bilder kam es zu einer falschen Zuordnung der Person.

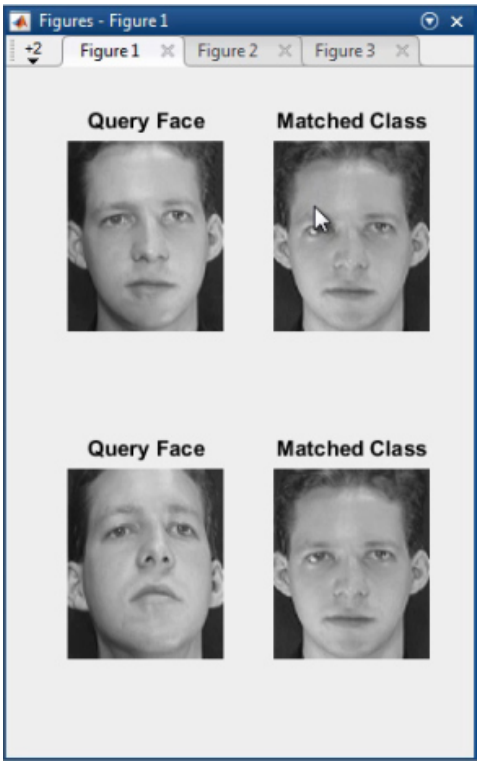

Abbildung 18: Matlab Face Recognition 1

Quelle: eigene Darstellung

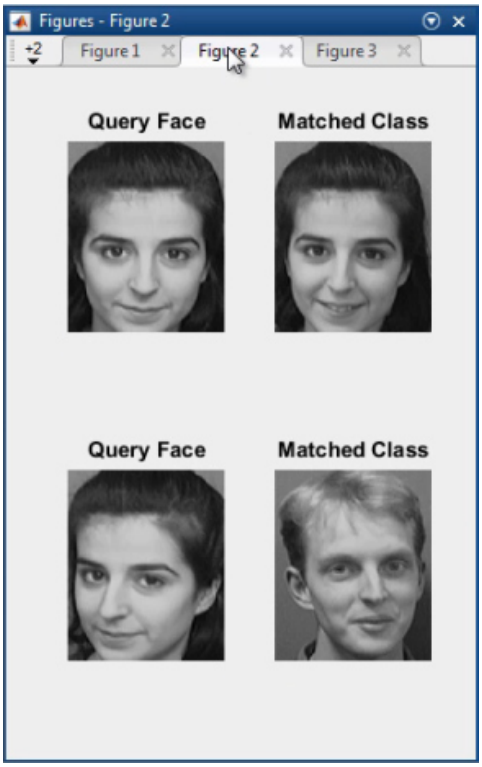

Abbildung 19: Matlab Face Recognition 2

Quelle: eigene Darstellung

Für den Benutzer bietet sich damit eine geeignete Anwendungsmöglichkeit an, die sich gut auf Veränderungen einstellen kann. Weiterhin sollte erwähnt werden, dass die Prozesse leicht optimiert werden können. Beispielsweise lässt sich der Code „for imgNum = 1:240“ durch die Zeile „parfor imgNum = 1:240“ ersetzten, um eine Parallelisierung und somit eine Bearbeitung über das Netzwerk zuzulassen. Dafür muss nur das vorher erstellte Cluster in Matlab verwendet werden. Alleine durch eine Parallelisierung lässt sich die Verarbeitungszeit von ungefähr 24 Minuten auf 1 Minute und 17 Sekunden reduzieren. Dieses Beispiel zeigt, wie leistungsstark Matlab mit seinen Zusatzfunktionen ist und dass dieses Programm für Face Detection in Betracht gezogen werden sollte.

3.5.3 Erkennung von Region of Interest (ROI)

Für diesen Test werden zwei Bilder miteinander verglichen. Das Query Image (links) soll mit einem Originalbild (rechts) überprüft werden. In diesem Test wird ein Holzelefant als Query Image verwendet.

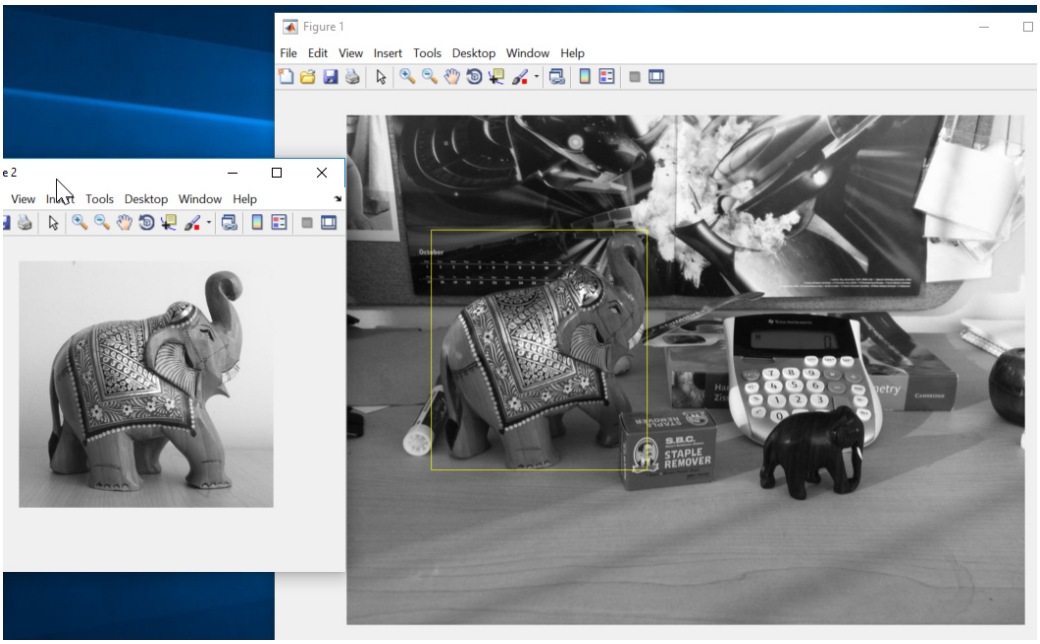

Abbildung 20: Matlab ROI 1

Quelle: eigene Darstellung

Die Erkennung im Originalbild funktioniert hervorragend. Obwohl hier eine Art Template Matching verwendet wird, kann das Query Image eindeutig identifiziert werden. Zum besseren Verständnis muss beachtet werden, dass bei einem Template Matching das gesuchte Bild keine Veränderungen gegenüber dem originalen Bild aufweisen darf, da es sonst nicht wiedererkannt werden kann. Obwohl ein weiterer Elefant mit ähnlicher Struktur rechts neben dem gesuchten Objekt steht, wird Letzteres richtig erkannt, womit eine viel höhere Wiedererkennung geboten wird.

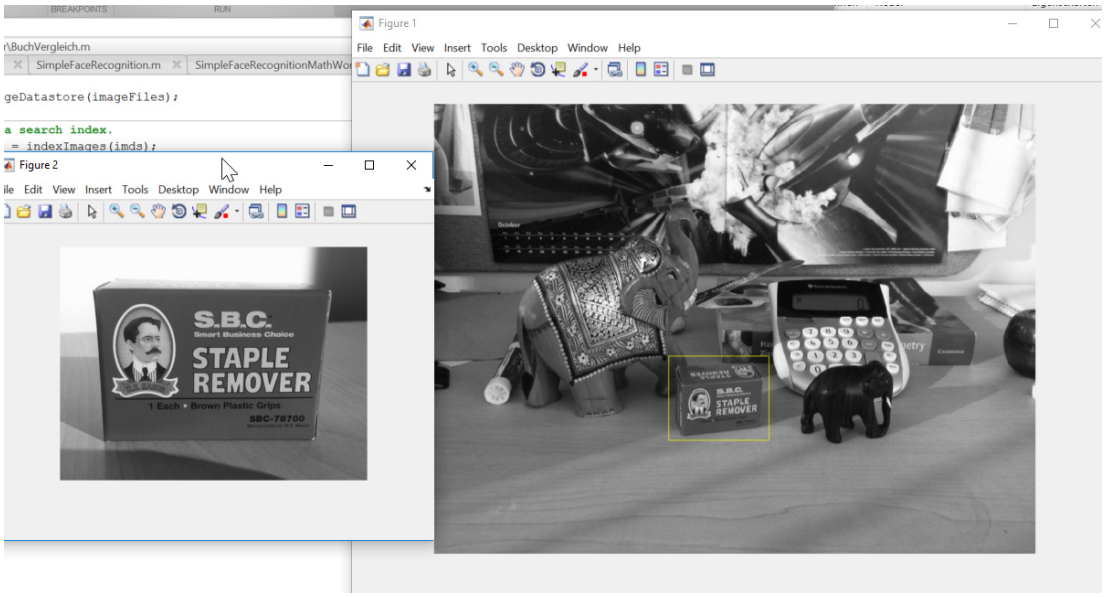

Abbildung 21: Matlab ROI 2

Quelle: eigene Darstellung

Im oben gezeigten Versuch wird als Test eine Verpackung eines Enthefters verwendet. Auch hier ist gut zu erkennen, dass das gesuchte Objekt richtig identifiziert wurde. Positiv an diesem Beispiel ist, dass das gesuchte Objekt im originalen Bild sogar eine leichte Drehung und einen anderen Winkel aufweist. Das Objekt kann trotz Veränderungen richtig wiedererkannt werden.

3.5.4 Bildvergleich

Im nächsten Test geht es um einen Buchvergleich. Es soll möglich sein, ein Query Image aus einer Reihe von 58 Bildern, die ähnliche Inhalte aufweisen, richtig zu erkennen. Die Schwierigkeit in diesem Fall liegt in der Drehung beim Query Image. Alle zu prüfenden Bilder sind entweder in einer normalen oder in einer gedrehten Lage wiederzuerkennen.

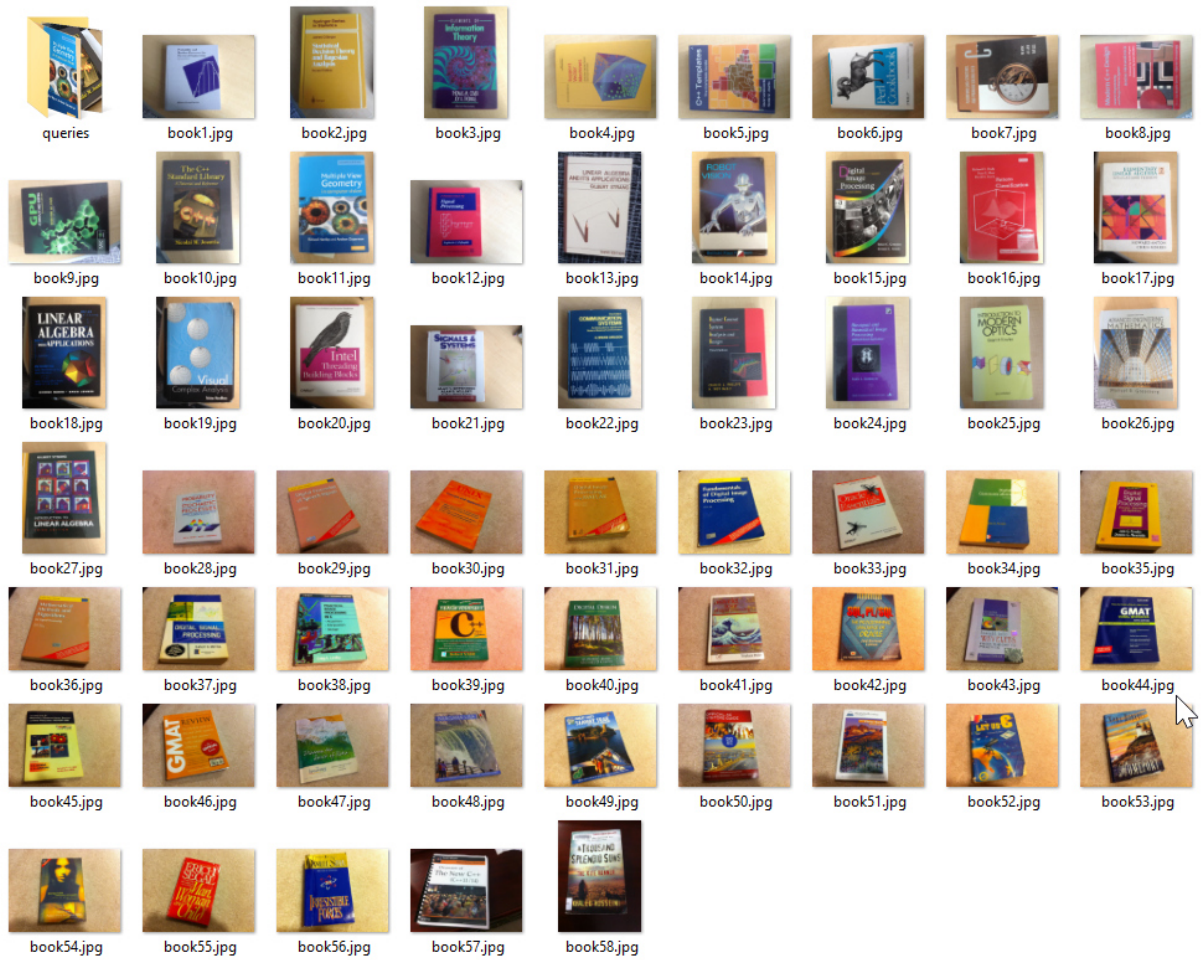

Abbildung 22: Matlab Buchtestset

Quelle: eigene Darstellung

Die oben aufgeführte Abbildung stellt das Testset dar. Anhand des Query Image in der Abbildung 23 wird das Testset überprüft.

Abbildung 23: Matlab Resultat Buchvergleich

Quelle: eigene Darstellung

Das Query Image wurde richtig erkannt und die Anwendung lässt so Raum für Variationen bei dem Originalbild.

3.5.5 Fazit Matlab

Matlab ist eine gute Software, die es dem Benutzer erlaubt, ohne große Programmierkenntnisse leistungsfähige Anwendungen zu schreiben. Zudem können durch Parallelisierung der Aufträge die vorhandenen Hardwareressourcen optimal genutzt werden. Die überzeugendsten Argumente für den Einsatz von Matlab sind die Benutzerfreundlichkeit und die dahinterstehende Community. Der relativ hohe Lizenzpreis in Höhe von 2.000,00 Euro steht dem Einsatz von Matlab entgegen.

3.6 Microsoft Computer Vision API

Das Unternehmen Microsoft bietet Entwicklern eine Reihe von Analysetools an. Microsoft unterteilt die Tools in Vision, Speech, Language, Knowledge und Search. Für die Tests in der vorliegenden Arbeit wird die Kategorie Vision Tools genauer betrachtet.[26]

3.6.1 Bildanalyse

Zunächst wird die Analyse der Bilder ausprobiert. Dieser Test lässt sich mit einem beliebigen Bild auf der Microsoft-Website durchführen.

Abbildung 24: Microsoft-Testbild[27]

Nach dem Hochladen des Bildes sind zahlreiche Informationen des Bildes erkennbar.

[26] Vgl. Computer Vision API. Online verfügbar unter https://www.microsoft.com/cognitive-services/en-us/computer-vision-api, zuletzt geprüft am 27.01.2017.
[27] Skyline. Online verfügbar unter https://portalstoragewuprod2.azureedge.net/vision/Analysis/2-1.jpg, zuletzt geprüft am 27.01.2017.

Tabelle 1: Bildanalyse

Feature Name	**Value**
Description	{ "type": 0, "captions": [{ "text": "a black and white photo of a large city", "confidence": 0.5984385403560294 }] }
Tags	[{ "name": **"sky",** "confidence": 0.9995020627975464 }, { "name": **"outdoor",** "confidence": 0.9976274371147156 }, { "name": **"city",** "confidence": 0.9237649440765381 }, { "name": **"white",** "confidence": 0.7209966778755188 }, { "name": **"skyscraper",** "confidence": 0.17128275334835052 }]
Image Format	JPEG
Image Dimensions	1500 x 1155
Clip Art Type	0 Non-clipart
Line Drawing Type	0 Non-LineDrawing
Black & White Image	True
Is Adult Content	False
Adult Score	0.009652587585151195
Is Racy Content	False
Racy Score	0.009652587585151196
Categories	[{ "name": **"building_"**, "score": 0.2734375 }, { "name": **"building_street"**, "score": 0.265625 }, { "name": **"outdoor_city",** "score": 0.25390625 }]
Faces	[]
Dominant Color Background	
Dominant Color Foreground	
Dominant Colors	
Accent Color	#303030

Relevante Features, die hier herausgefiltert werden, sind „Tags“, „Image Format“, „Image Dimensions“, „Categories“ und „Faces“. Mit diesen Features können schon einige Informationen über die Bilder gewonnen werden, ohne selbst viel Arbeit zu investieren.

3.6.2 Person Recognition

Ein nützliches Tool könnte die Erkennung von Prominenten sein.

Tabelle 2: Personenerkennung[28]

Abbildung 25: Satya Nadella

{ "requestId": "9db9db74-de37-4d9a-a6b6-c1b2ec2bdfb0",
"metadata": {
"height": 533,
"width": 800,
"format": "Jpeg" },
"result": {
"celebrities": [{
"name": "Satya Nadella",
"faceRectangle": {
"width": 134,
"height": 134,
"left": 401,
"top": 103 },
"confidence": 0.9576959 }] } }

Als Resultat werden der Name der Person, die Position des Gesichtes, die Auflösung des Bildes, das Geschlecht und in einigen Fällen auch das Alter der Person wiedergegeben. Diese Funktion ist hilfreich, wenn überprüft werden soll, ob es sich tatsächlich um eine bestimmte Person handelt und diese mit einer hohen Wahrscheinlichkeit identifiziert werden kann.

[28] Sataya Nadella. Online verfügbar unter https://tse4.mm.bing.net/th?id=OIP.M7567c257275aeaacd5e4037a5c69ae7do1&pid=Api, zuletzt geprüft am 27.01.2017.

3.6.3 Fazit Microsoft Computer Vision API

Microsoft bietet mit der Computer Vision API ein gutes Produkt mit guten und hilfreichen Features an. Positiv soll auch die umfangreiche und gut verständliche Dokumentation von Computer Vision API erwähnt werden. Dem Benutzer wird ein Beispielcode vorgegeben, der gut beschrieben wird und verständlich ist. Zudem gibt es noch eine Palette an weiteren Features der API. Der Preis für 1000 Anfragen liegt bei 1,50 $, sobald Benutzer mehr als 5000 Anfragen pro Monat setzen.[29]

3.7 Microsoft CNTK

Das CNTK (zukünftig Microsoft Cognitive Toolkit) von Microsoft ist eine Open-Source-Lösung, die unter anderem als Deep-Learning-Werkzeug zur Erkennung von Bildern und Sprache genutzt wird. Der Vorteile des Tools sind die schnelle Bearbeitung über das Netzwerk und die Verwendung der GPU. Ein weiterer Vorteil ist die Anwendung der Programmiersprachen C++ oder Python.[30]

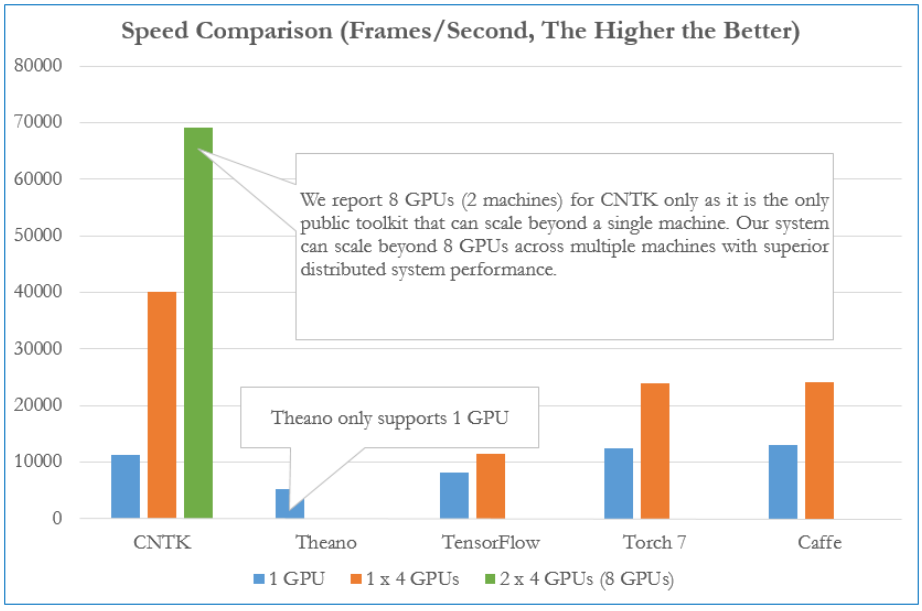

Abbildung 26: Deep Learning-Werkzeuge[31]

[29] Microsoft Computer Vision API Pricing options. Online verfügbar unter https://www.microsoft.com/cognitive-services/en-us/computer-vision-api, zuletzt geprüft am 15.02.2017.
[30] Vgl. Microsoft/CNTK. What is the Microsoft Cognitive Toolkit. Online verfügbar unter https://github.com/Microsoft/CNTK, zuletzt geprüft am 27.01.2017.
[31] Speed Comparison. Online verfügbar unter https://raw.githubusercontent.com/Microsoft/CNTK/master/Documentation/Documents/PerformanceChart.png, zuletzt geprüft am 27.01.2017.

In der Abbildung 27 ist der beachtliche Leistungsvorsprung gegenüber den anderen Deep-Learning-Werkzeugen deutlich zu sehen. Somit wäre ein beachtlicher Vorsprung mit dieser Technologie möglich.

3.7.1 CNTK-Objekterkennung

Um sich einen Überblick über die Leistung des Tools zu verschaffen, wird hier eine Beispielaufgabe bearbeitet. Dabei sollen Gegenstände in einem Kühlschrank wiedererkannt werden. Für dieses Beispiel werden Python-Skripte ausgeführt, die noch auf Ordnerstrukturen und Classifier angepasst werden müssen.

Da es sich hier um ein Beispiel handelt, werden drei Skripte nacheinander ausgeführt. Zunächst wird eine Textausgabe erstellt:

Tabelle 3: CTNK-Objekterkennung 1

Evaluation Detection	
AP for avocado	1.0000
AP for orange	0.2500
AP for butter	1.0000
AP for Champagne	1.0000
AP for eggBox	0.7500
AP for gerkin	1.0000
AP for joghurt	1.0000
AP for ketschup	1.0000
AP for orangeJuice	1.0000
AP for onion	1.0000
AP for pepper	0.7600
AP for tomato	0.6400
AP for water	0.5000
AP for milk	1.0000
AP for tabasco	1.0000
AP for mustard	1.0000
Mean Average Precision	0.8688

Beeindruckend ist die hohe Genauigkeit der erkannten Objekte in dem Bild. Mit einem weiteren Skript können die erkannten Gegenstände visuell anzeigt werden.

Abbildung 27: CNTK-Bilderkennung 1

Quelle: eigene Darstellung

Abbildung 28: CNTK-Bilderkennung 2

Quelle: eigene Darstellung

3.7.2 CNTK-eigene Objekterkennung

In diesem Beispiel sind die bereit gestellten Python-Skripte und die Bilder mit dem Inhalt des Kühlschrankes verwendet worden. Es wurde getestet, welche Genauigkeit erreicht werden kann, wenn alle Schritte selbstständig ausgeführt werden. Zunächst mussten die Gegenstände in den Bildern markiert werden. Dies wurde für jeden Gegenstand angewendet, der für die folgende Erkennung gewünscht wird. Als Nächstes mussten die markierten Gegenstände benannt werden.

Abbildung 29: CNTK-Bilderlabel

Quelle: eigene Darstellung

Sobald diese Schritte durchlaufen sind, werden die vorbereiteten Python-Skripte gestartet, um das Modell zu trainieren. Als Ergebnis wurden folgende Werte erhalten:

Tabelle 4: CNTK-Objekterkennung 2

Evaluation detection	
AP for avocado	0.6667
AP for orange	1.0000
AP for butter	1.0000
AP for Champagne	1.0000
AP for eggBox	1.0000
AP for gerkin	0.7500
AP for joghurt	1.0000
AP for ketschup	0.6667
AP for orangeJuice	0.7500
AP for onion	1.0000
AP for pepper	1.0000
AP for tomato	0.3867
AP for water	0.0278
AP for milk	1.0000
AP for tabasco	1.0000
AP for mustard	1.0000
Mean Average Precision	0.817

Bei identischen Bildern, die verwendet wurden, ist eine Abweichung der Genauigkeit von 0,043 % entstanden. Für den ersten selbstständigen Test wird dieses Ergebnis als überaus gut eingeordnet. Es gibt Gegenstände, die besser oder schlechter wiedererkannt wurden, was jedoch auf die menschliche Bearbeitung zurückzuführen ist.

3.7.3 Fazit CNTK

Das Toolkit von Microsoft kann eine beeindruckende Performance im Vergleich zu der Konkurrenz vorweisen: eine hohe Anpassung auf die Bedürfnisse des Benutzers und vorkonfigurierte Skripte, die dem Benutzer einen schnellen Einstieg erlauben. Zudem steht auch hinter CNTK eine Community, die sich gegenseitig unterstützt. Dies sollte in Zukunft Beachtung finden, da bei einem Troubleshooting auf Probleme anderer Benutzer zurückgegriffen werden kann. Ein weiterer positiver Punkt ist, dass es sich hierbei um einen Open-Source-Code handelt. Das bedeutet, dass keine Kosten anfallen, jedoch ein von Grund

auf leistungsfähiges Tool bereitgestellt wird. Zusätzlich kann das CNTK zukünftig auch für weitere Einsatzgebiete, beispielsweise in der Erkennung von Sprache, genutzt werden.

3.8 Tool vergleich

In diesem Abschnitt sollen die benutzten Werkzeuge miteinander verglichen werden, um den anschließenden Tooleinsatz für die Weiterarbeit zu entscheiden.

Tabelle 5: Toolvergleich

-	**Google API**	**Matlab**	**Microsoft API**	**CNTK**
Cost	+	2000€	+	Free
Language	C++, Java etc.	own language	C++, Java etc.	Python, BrainScript etc.
Test Exampels	+	+	+	+
Community	+	+	/	/
Free Code/Sharing	/	+	/	/
Good Dokumentation	/	+	- / +	+
Clustering	have to be created	+	have to be created	+
Use with GPU	/	/	/	+
Complex installation	-/+	-	-/+	-/+
User-friendly	-/+	+	-/+	/
Person recognition	-	/	+	/
Image	+	+	+	+
Text	+	+	+	+

Legend	
-	No
+	Yes
/	Unknown
	Important(good)

Gut zu erkennen ist, dass Microsoft mit CNTK den Benutzern ein leistungsfähiges Werkzeug anbietet. Die Vorteile liegen hier klar auf der Hand: keine monatlichen Kosten und vielfältige Optimierungsmöglichkeiten. Ein wesentliches Entscheidungskriterium ist auch die Verwendung einer oder mehrere GPUs. Somit lässt sich die Bearbeitungszeit deutlichverkürzen. Durch die Verbindung über das Netzwerk könnte somit die optimale Leistung erbracht werden. Eine gute Alternative wäre Matlab mit den zuvor genannten Erweiterungen. Jedoch wären die Kosten für die Software vergleichsweise hoch. Matlab sollte daher als zweite Möglichkeit in Betracht gezogen werden, da es ebenfalls ein leistungsfähiges Tool darstellt. Die APIs von Google und Microsoft bieten gute Features an, die sich je nach Verwendung auch kombinieren lassen. Gleichzeitig steigen die monatlichen Kosten in Abhängigkeit des Verwendungsgrades, was zu einem Risikofaktor werden kann. Ein komplett

anderer Gedankengang wäre eine Kombination aus unterschiedlichen Tools, um jeweils die besten Eigenschaften nutzen zu können. Das würde wiederum die Abhängigkeit steigern und weitaus höhere Kosten verursachen.

Damit ist die Entscheidung zunächst auf CNTK gefallen und die weitere Bearbeitung wird mit diesem Tool durchgeführt.

4 Evaluation des ersten Modells und Ergebnis

Nach dem vergangenen positiven Test wurde nun eine eigenständige Bilderkennung mit Nike-Logos ausprobiert. Dazu ist ein Ordner mit 19 Bildern und ein Testordner mit sechs Bildern erstellt worden. Die verwendeten Bilder hatten zunächst unterschiedliche Auflösungen und sind mit einem Skript auf eine Auflösung von 800x600 angepasst worden. Die Bilder sollten in drei Klassierungen unterteilt werden: „Nike“(Schriftzug), „Logo“ und „JustDoIt“(Schriftzug). Als Resultat kam eine durchschnittliche Genauigkeit von 0,0035 heraus (Wert 1,00 entspricht 100 %, 0,0 entspricht 0 %). Dies war ein absolut negatives Ergebnis, weshalb dieser Test wiederholt wurde. Dieses Mal wurde mit 200 Bildern gearbeitet. Die durchschnittliche Genauigkeit lag dann bei 0,3845 aufgeteilt in:

- Nike = 1.0000
- Logo = 0.1562
- JustDoIt = 0.0000

Dieses Ergebnis sollte mit Vorsicht betrachtet werden, da die Erkennung des Nike-Logos bei 100 % steht. Das könnte bedeuten, dass entweder die Erkennung optimal verläuft oder es unbekannte Probleme gibt. Dieser Test wurde nochmals mit zwei Klassifizierungen ausgeführt. In diesem Fall gab es nur die Klassifizierung in Nike mit einer Genauigkeit von 0.3333 und Logo mit einer Genauigkeit von 0.1905. Somit offenbarte sich, dass die Genauigkeit von 100 % bei Nike als falsch zu bewerten ist, da es beim zweiten Durchlauf keine Veränderungen bei dem Nike-Logo gab und nun eine Genauigkeit von 0.3333 % erzielt wurde. Dennoch war festzustellen, dass sich das Ergebnis verbessert hat. Mit einer wesentlichen Erhöhung der Anzahl der benutzten Bilder werden die zukünftigen Ergebnisse noch mehr an Qualität und Genauigkeit gewinnen.

Mit dem zu Verfügung gestellten Server und der Benutzung der CPU dauert die momentane Trainingsphase circa acht bis neun Stunden. Dies sollte vor dem Start der Trainingsphase beachten werden, da variable Tests zeitintensiv sind. Bei Nutzung eines dedizierten Servers könnten hier sicherlich deutliche Verbesserungen stattfinden. Auch der Einsatz von GPUs wäre für zukünftige Anwendungen von erheblichem Vorteil.

5 Resultierender Workflow

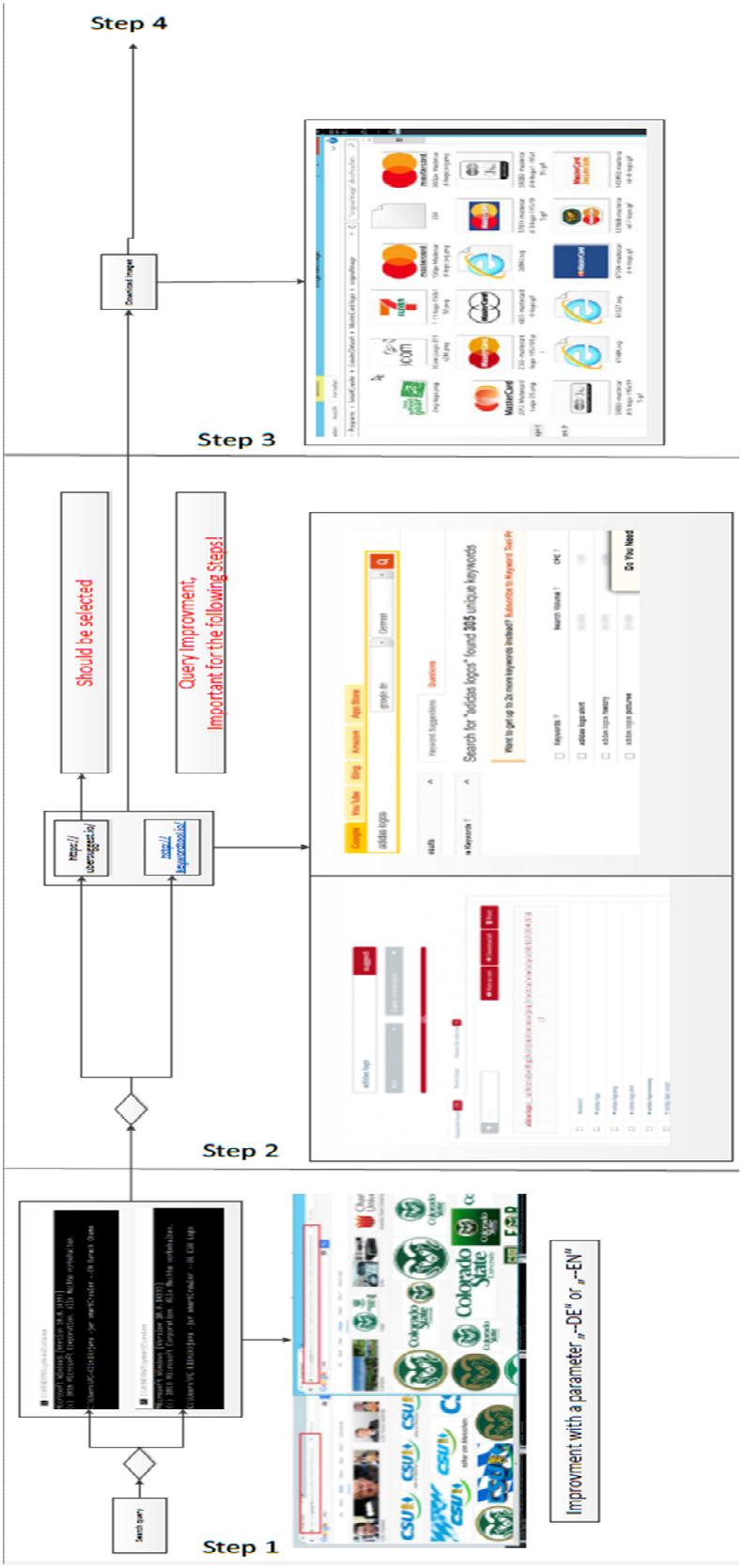

Abbildung 30: Resultierender Workflow 1

Quelle: eigene Darstellung

Abbildung 31: Resultierender Workflow 2

Quelle: eigene Darstellung

Nach dem Untersuchen der verschiedenen Tools war es sinnvoll, einen resultierenden Workflow zu gestalten. Der Workflow ist zunächst in neun Schritte unterteilt, die nachfolgend dargestellt werden.

Schritt 1: Search Query

Im ersten Schritt soll der Benutzer den Suchbegriff eingeben. Als Optimierungsmöglichkeit könnte ein Anhang an den Befehl gegeben werden, um die Suchregion zu bestimmen, beispielsweise „CDU logo –DE“, um die Suche primär auf die deutsche Region anzupassen.

Schritt 2: Keyword Website

Der SmartCrawler bezieht seine Keywords von der Website, um die Weite des Suchbegriffs zu vergrößern. Optimierungspotenzial wäre hier, die angesteuerte Website zu ändern und somit die erste Suchanfrage zu manipulieren.

Schritt 3: Download Images

In diesem Schritt werden die Bilder geladen und auf dem Server in ein zuvor erstelltes Verzeichnis gespeichert.

Schritt 4: Set ROIs

Schritt vier ist dafür gedacht, um auf einem geladenen Bild ein ROI zu setzen und die übrigen Bilder darauf zu testen. Das Bild mit dem ROI sollte als Vergleich dienen, um falsche Bilder zu selektieren.

Schritt 5: Create Folders

Hier wird die Verzeichnisstruktur aufgebaut. Als Oberbegriff wird der Name der Suchanfrage benutzt und darunter werden drei weitere Ordner mit den Namen: „negative“, „positive“, „testImage“ verwendet.

Schritt 6: Copy Image

Bei einer positiven Überprüfung im Schritt vier soll das Bild in das erstellte Verzeichnis „positive“ kopiert werden.

Schritt 7: Setup Folders

Nun werden die restlichen beiden Ordner mit Bildern befüllt. Theoretisch kann der Ordner „negative“ leer gelassen werden. In den Ordner „testImages“ können unterschiedliche Bilder geladen werden.

Schritt 8: Training etc.

Das Modell wird auf die geladenen Bilder trainiert. Dieser Schritt kann je nach Gestaltung einige Stunden in Anspruch nehmen.

Schritt 9: Analyze Images

Im letzten Schritt sollten die Bilder und das Ergebnis analysiert werden, um mögliche Verbesserungen für die Zukunft zu erhalten.

6 Fazit

Ziel dieser Arbeit war, die bestehenden Fehlerquellen im SmartCrawler-Prozess zu finden und diese mit der dahinterstehenden Problematik aufzuzeigen. Weiterhin ist die Vielfältigkeit der entstehenden Fehler präsentiert und mit Beispielen belegt worden. Anschließend wurden unterschiedliche Tools auf ihre Einsatzmöglichkeiten getestet. Die getesteten Tools zeigten auf, welche Leistung erbracht werden konnte und ob sie für den folgenden Workflow geeignet sind. In einem sodann folgenden Vergleich wurden die Stärken sowie die Schwächen der getesteten Tools aufgelistet und miteinander verglichen. Als resultierender Sieger des Vergleichs wurde CNTK von Microsoft gewählt. Durch das große Leistungspotenzial dieses Tools könnten in Zukunft erhebliche Leistungsvorsprünge erbracht werden. Nach Abschluss der Analyse und der Auswahl der zu nutzenden Tools wurde ein Workflow erstellt. Dieser Workflow soll darstellen, wie das Training eines Modells zustande kommt, um Logos bzw. Personen wiederzuerkennen. In dem Workflow befinden sich auch Verbesserungspotenziale für den SmartCrawler, der die Suche durch bestimmte Kriterien verfeinern könnte. Anhand des Workflows kann Schritt für Schritt nachvollzogen werden, wie der komplette Ablauf aufgebaut wird.

Abschließend kann gesagt werden, dass entstehende Fehlerquellen beim SmartCrawler-Prozess erkannt und mögliche Verbesserungen erkannt oder benannt wurden. Der SmartCrawler-Prozess ist essenziell für das anschließende Training. Somit richtet sich das Augenmerk auf diesen Prozess, da es ansonsten zu erheblichen Genauigkeitsverlusten bei dem Modell kommt. Bei der Verwendung von CNTK sollten die Vorteile dieses Tools gegenüber den konkurrierenden Tools erkannt und eingesetzt werden. Erst dadurch werden die Vorteile von CNTK gegenüber anderen Tools deutlich sichtbar. Weiterhin kann CNTK auch für andere Einsatzgebiete genutzt werden. Somit steht dem Benutzer ein vielfältiges Tool zur Verfügung, das zeitgleich zukunftssicher gestaltet ist.

Der entstandene Workflow kann zukünftig auch durch andere Methoden erweitert oder Teile davon optimiert werden. Somit können einzelne Komponenten aus dem Workflow gegebenenfalls ausgetauscht und durch effizientere Komponenten ersetzt werden. Ein Einsatz der vorgestellten Tools in dem Workflow oder das Auslagern der Anwendung auf leistungsstärkere Maschinen kann ebenfalls in Betracht gezogen werden. Vorteilhaft wäre die Verwendung eines dezidierten Computerclusters. Ausschlaggebend ist die Nutzung des Potenzials der angewandten Tools, damit daraus eine optimale Leistung resultiert. Ebenfalls sollte getestet werden, ob CNTK in weiteren Anwendungsgebieten Einzug finden könnte. Durch die vielfältigen Einsatzmöglichkeiten von CNTK könnte dieses Tool auch in anderen

Anwendungsbereichen wie Konsumanalyse, Qualitätskontrolle oder bei der Erkennung von Mustern einen erheblichen Vorteil verschaffen.

Literaturverzeichnis

Abts, Dietmar (2016): Grundkurs JAVA. Von den Grundlagen bis zu Datenbank- und Netzanwendungen. 9., überarbeitete und erweiterte Auflage. Wiesbaden: Springer Vieweg (Lehrbuch).

Adidas Logo. Online verfügbar unter https://upload.wikimedia.org/wikipedia/commons/thumb/d/d4/Adidas-logo.svg/2000px-Adidas-logo.svg.png, zuletzt geprüft am 27.01.2017.

Advances in Face Detection and Facial Image Analysis (2016). Cham: Springer International Publishing.

Bill Gates Speech. Online verfügbar unter http://3.bp.blogspot.com/-HllDa0woDpY/UfAsz0KTfvI/AAAAAAAAAlw/UZ503x1eYIY/s1600/Bill+Gates+Microsoft+Will+Not+Return.jpg, zuletzt geprüft am 27.01.2017.

Cloud Vision API. Online verfügbar unter https://cloud.google.com/vision/, zuletzt geprüft am 27.01.2017.

Computer Vision API. Online verfügbar unter https://www.microsoft.com/cognitive-services/en-us/computer-vision-api, zuletzt geprüft am 27.01.2017.

Garcia, Gloria Bueno; Gracia, Ismael Serrano; Aranda, Jose Lius Espinosa; Salido Tercero, Jesus; Enano, Noelia Vallez; Suarez, Oscar Deniz (2015): Learning image processing with OpenCV. Exploit the amazing features of OpenCV to create powerful image processing applications through easy-to-follow examples. Birmingham, UK: Pakct Publishing (Community experience distilled).

Google Cloud Vision API Preisliste. Online verfügbar unter https://cloud.google.com/vision/pricing, zuletzt geprüft am 15.02.17.

Görz, Günther; Schneeberger, Josef; Schmid, Ute (2014): Handbuch der künstlichen Intelligenz. 5., überarb. und aktualisierte Aufl. München: Oldenbourg.

Java Timeline. Online verfügbar unter http://oracle.com.edgesuite.net/timeline/java/, zuletzt geprüft am 10.02.2017.

Katze. Online verfügbar unter http://klaus-von-gierke.de/wp-content/uploads/Katze-Senior.jpg, zuletzt geprüft am 27.01.2017.

Kruse, Rudolf; Borgelt, Christian; Braune, Christian; Klawonn, Frank; Moewes, Christian; Steinbrecher, Matthias (2015): Computational Intelligence. Eine methodische

Einführung in künstliche neuronale Netze, evolutionäre Algorithmen, Fuzzy-Systeme und Bayes-Netze. 2., überarbeitete und erweiterte Auflage. Wiesbaden: Springer Vieweg (Computational Intelligence).

Lutz, Mark; Schulten, Lars (2010): Python. Kurz & gut. 4. Aufl. Köln: O'Reilly (O'Reillys Taschenbibliothek).

Matlab. Online verfügbar unter https://de.mathworks.com/products/matlab.html, zuletzt geprüft am 27.01.2017.

Matlab. Online verfügbar unter https://i.ytimg.com/vi/pRsGM7H91VY/maxresdefault.jpg, zuletzt geprüft am 27.01.2017.

Microsoft Computer Vision API Pricing options. Online verfügbar unter https://www.microsoft.com/cognitive-services/en-us/computer-vision-api, zuletzt geprüft am 15.02.2017.

Microsoft/CNTK. What is the Microsoft Cognitive Toolkit. Online verfügbar unter https://github.com/Microsoft/CNTK, zuletzt geprüft am 27.01.2017.

Mitchell, Tom M. (20]10): Machine learning. International ed., [Reprint.]. New York, NY: McGraw-Hill (McGraw-Hill series in computer science).

Nike Logo. Online verfügbar unter http://news.nike.com/, zuletzt geprüft am 27.01.2017.

Sataya Nadella. Online verfügbar unter https://tse4.mm.bing.net/th?id=OIP.M7567c257275aeaacd5e4037a5c69ae7do1&pid=Api, zuletzt geprüft am 27.01.2017.

Skyline. Online verfügbar unter https://portalstoragewuprod2.azureedge.net/vision/Analysis/2-1.jpg, zuletzt geprüft am 27.01.2017.

Speed Comparison. Online verfügbar unter https://raw.githubusercontent.com/Microsoft/CNTK/master/Documentation/Documents/PerformanceChart.png, zuletzt geprüft am 27.01.2017.

Springer Gabler Verlag (Hg.): Definition » maschinelles Lernen « | Gabler Wirtschaftslexikon. Online verfügbar unter http://wirtschaftslexikon.gabler.de/Definition/maschinelles-lernen.html, zuletzt geprüft am 27.01.2017.

Viola, Paul; Jones, Michael J. (2004): Robust Real-Time Face Detection. In: *International Journal of Computer Vision* 57 (2), S. 137–154. DOI: 10.1023/B:VISI.0000013087.49260.fb.

Eidesstattliche Erklärung

Ich versichere, dass ich die vorstehende Arbeit selbständig verfasst und hierzu keine anderen als die angegebenen Hilfsmittel verwendet habe. Alle Stellen der Arbeit die wörtlich oder sinngemäß aus fremden Quellen entnommen wurden, sind als solche kenntlich gemacht.

Die Arbeit wurde bisher in gleicher oder ähnlicher Form in keinem anderen Studiengang als Prüfungsleistung vorgelegt oder an anderer Stelle veröffentlicht.

Ich bin mir bewusst, dass eine falsche Erklärung rechtliche Folgen haben kann.

Albbruck, 20.02.17, Alexander Gerling

Danksagung

An dieser Stelle möchte ich mich bei all denjenigen bedanken, die mich während der Anfertigung dieser Bachelorarbeit unterstützt und motiviert haben.

Zuerst gebührt mein Dank Herrn Prof. Dr. Andreas Hess und Herrn Dr. Patrick Ndjiki-Nya, die meine Bachelorarbeit betreut und begutachtet haben. Für die hilfreichen Anregungen und die konstruktive Kritik bei der Erstellung dieser Arbeit möchte ich mich herzlich bedanken.

Ebenfalls möchte ich mich bei den Mitarbeitern von PAMA Technologies bedanken, die mir mit viel Geduld, Interesse und Hilfsbereitschaft zur Seite standen. Ganz besonders bei Herrn Chavan, der sich immer um meine Anliegen gekümmert hat.

Meinen Freunden Andrius Oliveira, Eugen Jesipow, Familie Gampp und ganz besonders meiner Freundin Annabella Gampp danke ich besonders für den starken emotionalen Rückhalt über die Dauer meines gesamten Studiums.

Abschließend möchte ich mich bei meiner Familie bedanken, die mir mein Studium durch ihre Unterstützung erst ermöglichten und stets ein offenes Ohr für meine Sorgen hatten.

Alexander Gerling

Albbruck, 20.02.2017

Anhang

Alle relevanten Daten, finden Sie digital auf der mitgeführten CD.